

日 本 国 特 許 庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日            2 0 0 3 年   7 月 2 2 日  
Date of Application:

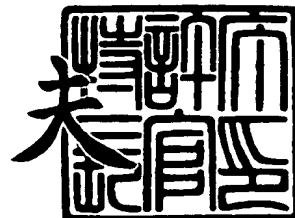
出 願 番 号            特 願 2 0 0 3 - 1 9 9 9 4 9  
Application Number:  
[ST. 10/C]:            [ J P 2 0 0 3 - 1 9 9 9 4 9 ]

出   願   人            株 式 会 社 リ コ ー  
Applicant(s):

2 0 0 3 年   8 月   4 日

特許庁長官  
Commissioner,  
Japan Patent Office

今 井 康 夫



【書類名】 特許願

【整理番号】 0305121

【提出日】 平成15年 7月22日

【あて先】 特許庁長官 今井 康夫 殿

【国際特許分類】 G03G 21/00 370

【発明の名称】 情報処理装置およびそのプログラム

【請求項の数】 23

【発明者】

    【住所又は居所】 東京都大田区中馬込1丁目3番6号 株式会社リコー内

    【氏名】 大石 勉

【発明者】

    【住所又は居所】 東京都大田区中馬込1丁目3番6号 株式会社リコー内

    【氏名】 安藤 光男

【発明者】

    【住所又は居所】 東京都大田区中馬込1丁目3番6号 株式会社リコー内

    【氏名】 杉浦 裕子

【特許出願人】

    【識別番号】 000006747

    【氏名又は名称】 株式会社リコー

【代理人】

    【識別番号】 100070150

    【弁理士】

    【氏名又は名称】 伊東 忠彦

【先の出願に基づく優先権主張】

    【出願番号】 特願2002-218814

    【出願日】 平成14年 7月26日

【先の出願に基づく優先権主張】

    【出願番号】 特願2002-295378

    【出願日】 平成14年10月 8日

## 【手数料の表示】

【予納台帳番号】 002989

【納付金額】 21,000円

## 【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9911477

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 情報処理装置およびそのプログラム

【特許請求の範囲】

【請求項 1】 画像形成装置で実行されるプログラムを開発するために使用される情報処理装置であって、

プログラムを解釈し、実行する実行手段と、

当該プログラムの命令に対する画像形成装置の動作に相当する動作を前記情報処理装置に実行させるエミュレータと

を備えたことを特徴とする情報処理装置。

【請求項 2】 前記実行手段は仮想マシンである請求項 1 に記載の情報処理装置。

【請求項 3】 ソースコードを、前記仮想マシンが実行可能なコードに変換するコンパイラを更に備えた請求項 2 に記載の情報処理装置。

【請求項 4】 前記プログラムは J a v a（登録商標）プログラムであり、当該 J a v a（登録商標）プログラムが参照するクラスライブラリを更に備えた請求項 3 に記載の情報処理装置。

【請求項 5】 前記エミュレータは、前記情報処理装置に、前記画像形成装置の操作部に相当する画面を表示させる請求項 1 に記載の情報処理装置。

【請求項 6】 前記エミュレータは、前記情報処理装置に、プログラムのリストを表示させ、ユーザにより選択されたプログラムを実行する請求項 1 に記載の情報処理装置。

【請求項 7】 前記エミュレータは、印刷命令に対して、前記情報処理装置に、当該印刷命令で要求されたデータを当該情報処理装置に接続されたプリンタに印刷させる請求項 1 に記載の情報処理装置。

【請求項 8】 前記情報処理装置は、作成されたプログラムを送信する送信手段を更に備える請求項 1 ないし 7 のうちいずれか 1 項に記載の情報処理装置。

【請求項 9】 前記送信手段は、プログラムを格納するサーバに前記作成されたプログラムを送信する請求項 8 に記載の情報処理装置。

【請求項 10】 前記送信手段は、前記画像形成装置からプログラム入力画

面データを受信し、当該プログラム入力画面データを用いて前記プログラムを画像形成装置に送信する請求項 8 に記載の情報処理装置。

【請求項 1 1】 画像形成装置で実行されるアプリケーションプログラムを開発するために使用される情報処理装置を、

アプリケーションプログラムの命令に対する画像形成装置の動作に相当する動作を前記情報処理装置に実行させるエミュレータとして機能させるプログラム。

【請求項 1 2】 前記エミュレータは、前記アプリケーションプログラムを解釈し、実行する実行手段を含む請求項 1 1 に記載のプログラム。

【請求項 1 3】 前記実行手段は仮想マシンである請求項 1 2 に記載のプログラム。

【請求項 1 4】 ソースコードを前記仮想マシンが実行可能なコードに変換するコンパイラを含む請求項 1 3 に記載のプログラム。

【請求項 1 5】 前記アプリケーションプログラムは J a v a（登録商標）プログラムであり、当該 J a v a（登録商標）プログラムが参照するクラスライブラリを更に備えた請求項 1 4 に記載のプログラム。

【請求項 1 6】 前記エミュレータは、前記情報処理装置に、前記画像形成装置の操作部に相当する画面を表示させる請求項 1 1 に記載のプログラム。

【請求項 1 7】 前記エミュレータは、前記情報処理装置に、アプリケーションプログラムのリストを表示させ、ユーザにより選択されたアプリケーションプログラムを実行する請求項 1 1 に記載のプログラム。

【請求項 1 8】 前記エミュレータは、印刷命令に対して、前記情報処理装置に、当該印刷命令で要求されたデータを当該情報処理装置に接続されたプリンタに印刷させる請求項 1 1 に記載のプログラム。

【請求項 1 9】 前記情報処理装置を、作成されたプログラムを送信する送信手段として更に機能させる請求項 1 1 ないし 1 8 のうちいずれか 1 項に記載のプログラム。

【請求項 2 0】 前記送信手段は、プログラムを格納するサーバに前記作成されたプログラムを送信する請求項 1 9 に記載のプログラム。

【請求項 2 1】 前記送信手段は、前記画像形成装置からプログラム入力画

面データを受信し、当該プログラム入力画面データを用いて前記プログラムを画像形成装置に送信する請求項 1 9 に記載のプログラム。

【請求項 2 2】 請求項 1 ないし 1 0 のうちいずれか 1 項に記載の情報処理装置を用いて開発されたプログラム。

【請求項 2 3】 請求項 1 1 ないし 2 2 のうちいずれか 1 項に記載のプログラムを記録したコンピュータ読み取り可能な記録媒体。

#### 【発明の詳細な説明】

##### 【0 0 0 1】

##### 【発明の属する技術分野】

この発明は、アプリケーションプログラムを開発するための情報処理装置とそのためのプログラムに関するものである。

##### 【従来の技術】

近年では、プリンタ、コピー、ファクシミリ、スキャナなどの各装置の機能を 1 つの筐体内に収納した画像形成装置（以下、「複合機」という。）が知られている。この複合機は、1 つの筐体内に表示部、印刷部および撮像部などを設けるとともに、プリンタ、コピーおよびファクシミリ装置にそれぞれ対応した 3 種類のソフトウェアを設け、これらのソフトウェアを切り替えることによって、当該装置をプリンタ、コピー、スキャナ又はファクシミリ装置として動作させるものである。

##### 【0 0 0 2】

このような従来の複合機では、利用者に対し複合機の各種操作を行わせるためのオペレーションパネルが設けられ、このオペレーションパネルに操作画面の表示およびタッチ操作入力を行わせるための操作表示部を表示している。従来の複合機では、予め提供される機能が定められていたため、利用者が複合機に対して行う操作も大きな変更はなく、操作表示部の画面表示やタッチ操作による動作をカスタマイズする必要性はない。このため、従来の複合機には操作表示部のカスタマイズ機能は搭載されていなかった。

##### 【0 0 0 3】

##### 【特許文献 1】

特開 2002-152446 号公報

【0004】

【発明が解決しようとする課題】

ところで、このような従来の複合機では、プリンタ、コピー、スキャナおよびファクシミリ装置に対応するソフトウェアをそれぞれ別個に設けているため、各ソフトウェアの開発に多大の時間を要する。このため、出願人は、表示部、印刷部および撮像部などの画像形成処理で使用されるハードウェア資源を有し、プリンタ、コピー又はファクシミリなどの各ユーザサービスにそれぞれ固有の処理を行うアプリケーションを複数搭載し、これらのアプリケーションとハードウェア資源との間に介在して、ユーザサービスを提供する際に、アプリケーションの少なくとも2つが共通に必要とするハードウェア資源の管理、実行制御並びに画像形成処理を行う各種コントロールサービスからなるプラットフォームを備えた画像形成装置（複合機）を発明した。なお、各種コントロールサービスが行う処理をシステム側の処理と呼ぶ。

【0005】

このような新規な複合機では、アプリケーションと、ハードウェア資源にアクセスするような開発が難しい処理を行うコントロールサービスとを別個に設けているため、複合機の出荷後にユーザもしくは第三者であるサードベンダが画像形成処理などにかかるアプリケーションとして新規な外部アプリケーションを開発して複合機に搭載可能な構成となっている。

【0006】

このため、新たに開発された外部アプリケーションの起動や実行中に操作表示部における画面表示やタッチ操作によって、予め提供されている機能とは異なる実行処理を行わせたい場合がある。このように、新規な複合機では、操作表示部のカスタマイズという、出荷後に外部アプリケーションを搭載することを想定していない従来の複合機では問題にならなかった新規な課題が生じてくる。

【0007】

このような操作表示部のカスタマイズを、外部アプリケーションの開発段階でソースコードを記述し、ソースコードをコンパイルおよびリンクすることによっ

て行うことが考えられる。すなわち、外部アプリケーションの開発段階で、外部アプリケーション自体の処理の中に操作表示部のタッチ操作に伴う実行処理を開発する場合が多くなる。この場合、操作表示部のタッチ操作に伴う実行処理の検証を逐次行いながら外部アプリケーション自体の開発を進めていくことがプログラム開発効率上好ましい。しかしながら、タッチ操作に伴う実行処理に障害を検出するたびに、ソースコードの修正、再コンパイル、再リンクを行って、開発対象の複合機に再インストールし、動作検証を行わなければならないとすると、プログラム開発の効率が悪くなるという問題がある。

#### 【0008】

また、操作表示部のカスタマイズに限らず、複合機とは別のPC等の環境でのプログラムの作成と、複合機での動作検証とによる従来のプログラム開発は効率が悪いという問題がある。

#### 【0009】

この発明は上記に鑑みてなされたもので、アプリケーションプログラム開発効率を向上させることができる情報処理装置およびプログラムを得ることを目的とする。

#### 【課題を解決するための手段】

上記の課題は以下の発明によって解決できる。

#### 【0010】

請求項1に記載の発明は、画像形成装置で実行されるプログラムを開発するために使用される情報処理装置であって、プログラムを解釈し、実行する実行手段と、当該プログラムの命令に対する画像形成装置の動作に相当する動作を前記情報処理装置に実行させるエミュレータとを備える。

#### 【0011】

本発明によれば、実行手段により実行されたプログラムによりなされる画像形成装置に対する命令に対する動作に相当する動作を、エミュレータが情報処理装置に対して実行するので、情報処理装置上で、プログラムの動作確認を行うことができ、プログラムの開発効率が向上する。

#### 【0012】



請求項 2 に記載の発明は、請求項 1 の記載において、前記実行手段は仮想マシンであるとするものである。仮想マシンを搭載することにより、J a v a（登録商標）プログラムを実行できる。

【0013】

請求項 3 に記載の発明は、請求項 2 に記載の情報処理装置が、ソースコードを前記仮想マシンが実行可能なコードに変換するコンパイラを更に備えるものである。コンパイラにより、ソースコードをバイトコードに変換できる。

【0014】

請求項 4 に記載の発明は、請求項 3 の記載において、前記プログラムは J a v a（登録商標）プログラムであり、情報処理装置が当該 J a v a（登録商標）プログラムが参照するクラスライブラリを更に備えるものである。クラスライブラリを備えることにより、種々の機能を提供する J a v a（登録商標）プログラムを開発できる。

【0015】

請求項 5 に記載の発明は、請求項 1 の記載において、前記エミュレータは、前記情報処理装置に、前記画像形成装置の操作部に相当する画面を表示させる。

【0016】

本発明により、情報処理装置上で画像形成装置の表示部における操作と同様の操作を行うことができる。

【0017】

請求項 6 に記載の発明は、請求項 1 の記載において、前記エミュレータは、前記情報処理装置に、プログラムのリストを表示させ、ユーザにより選択されたプログラムを実行するものである。

【0018】

本発明によれば、ユーザは所望のプログラムを実行させてプログラムのテストを行うことができる。

【0019】

請求項 7 に記載の発明は、請求項 1 の記載において、前記エミュレータは、印刷命令に対して、前記情報処理装置に、当該印刷命令で要求されたデータを当該

情報処理装置に接続されたプリンタに印刷させるものである。

【0020】

本発明によれば印刷命令の確認を行うことができる。

【0021】

請求項 8 に記載の発明は、請求項 1 ないし 7 のうちいずれか 1 項に記載の情報処理装置が、作成されたプログラムを送信する送信手段を更に備えるというものである。

【0022】

本発明によれば、作成されたプログラムを例えば画像形成装置に送信できる。

【0023】

また、請求項 9 に記載の発明は、請求項 8 の記載において、前記送信手段は、プログラムを格納するサーバに前記作成されたプログラムを送信するというものである。

【0024】

上記の発明によれば、上記のサーバにプログラムを格納できるので、画像形成装置がサーバからプログラムをダウンロードできる。

【0025】

請求項 10 に記載の発明は、請求項 8 の記載において、前記送信手段は、前記画像形成装置からプログラム入力画面データを受信し、当該プログラム入力画面データを用いて前記プログラムを画像形成装置に送信するものである。

【0026】

本発明によれば、画像形成装置から受信したプログラム入力画面データを用いるので、容易にプログラムを送信することができる。

【0027】

請求項 11 ～ 21 に記載の発明は、画像形成装置のエミュレータを含むプログラムの発明である。また、請求項 21 に記載の発明は、上記の情報処理装置を用いて開発されたプログラムの発明である。請求項 23 に記載の発明はプログラムを記録した記録媒体の発明である。

【0028】

**【発明の実施の形態】**

以下に添付図面を参照して、この発明にかかる画像形成装置および操作表示部プログラム生成方法の好適な実施の形態を詳細に説明する。

**【0029】****(実施の形態1)**

図1は、この発明の実施の形態1である画像形成装置（以下、「複合機」という）の主要部の構成とネットワーク構成を示すブロック図である。実施の形態1にかかる複合機100は、インターネットに接続され、当該インターネットに接続されたPC（Personal Computer）などのクライアント端末から複合機の操作を行うオペレーションパネルの操作表示部のカスタマイズプログラムの入力およびデバッグ作業を行うことを可能としたものである。

**【0030】**

図1に示すように、複合機100とPC200は、インターネット220で接続されており、通信プロトコルとしてTCP/IPを利用している。複合機100は、複合機初期化部129と、プログラム起動部131と、プログラミングサービス132と、インタプリタ134と、NCS（ネットワークコントロールサービス）128と、httpd（httpデーモン）106と、共有メモリ105と、HDD103とを備えた構成となっている。

**【0031】**

プログラミングサービス132は、ネットワークに接続されたクライアント端末としてのPC200のWEBブラウザに後述するhttpファイルで構成されたWEBページであるプログラミング画面201を表示し、このプログラミング画面201からオペレーションパネル210の操作表示部のカスタマイズプログラムを入力させるものである。また、プログラミングサービス132は、PCから入力されたカスタマイズプログラムをHDD103のカスタマイズ領域として作成されたカスタマイズディレクトリ212に保存する。さらに、プログラミングサービス132は、複合機100上でカスタマイズプログラムが実行されるように、後述する起動設定ファイル211に入力されたカスタマイズプログラムを登録する。

**【0032】**

ここで、プログラミングサービス132とPC200のWEBブラウザの間のプログラミング画面などのhttpファイルの転送は、httpプロトコルに従って行われる。このため、プログラミングサービス132は、Webサーバ（httpサーバ）としての役割も担っている。

**【0033】**

カスタマイズプログラムは、インタプリタで解釈可能な言語で記述され、具体的には、汎用OSがUNIX（登録商標）の場合、シェルスクリプトである。

**【0034】**

インタプリタ134は、カスタマイズプログラムをステップごとに解析し逐次実行するものであり、本実施の形態では、シェルスクリプトを実行するシェル（bsh, cshなど）である。このシェル134は、汎用OS121としてのUNIX（登録商標）にOSのカーネル135とは別個に存在するものである。

**【0035】**

プログラム起動部131は、HDD103の診断結果が正常である場合に、HDD103にインストールされている外部アプリ117を起動するものである。また、プログラム起動部131は、ICカードなどの記憶媒体に保存された外部アプリ117を起動する。

**【0036】**

複合機初期化部129は、汎用OS121の上で最初に起動されるプロセスであり、コントロールサービスやアプリケーション130（外部アプリ117を除く）の起動およびプログラム起動部131の起動を行うものである。

**【0037】**

NCS128は、ネットワークを制御するものであり、httpd106から通知されたリクエストメッセージ受信の旨をプログラミングサービス132に通知する。

**【0038】**

httpd106は、OSに含まれるプロセス（デーモン）であり、ポート80番を常時監視してリクエストメッセージの受信を行うとともに、レスポンスメ

ッセージの送信を行うものである。なお、リクエストメッセージおよびレスポンスメッセージの構造は、通常の http プロトコルにおける各メッセージの構造と同様であり、各メッセージには、html 形式で記述されたメッセージボディが含まれている。また、httpd 106 は、インターネット 220 経由で受信したリクエストメッセージ受信の旨を NCS 128 に通知するとともに、リクエストメッセージを共有メモリ 105 に格納する。

#### 【0039】

この共有メモリ 105 は、httpd 106 とプログラミングサービス 132 とのプロセス間通信に利用されるものであり、共有メモリ 105 を介してリクエストメッセージ、レスポンスメッセージなどの受け渡しを行う。

#### 【0040】

HDD 103 には、起動設定ファイル 211 が格納され、また HDD 103 のカスタマイズディレクトリ 212 にカスタマイズプログラムが格納される。

#### 【0041】

PC 200 は、複合機 100 のプログラミングサービス 132 をサーバとしたクライアント端末であり、PC 200 で実行される WEB ブラウザにプログラミングサービス 132 から http プロトコルで送信されたプログラミング画面 201 やその他画面を表示し、操作入力を行えるように構成されている。

#### 【0042】

次に、本実施の形態にかかる複合機 100 の全体の機能的構成について説明する。図 2 は、実施の形態 1 の複合機 100 の構成を示すブロック図である。図 2 に示すように、複合機 100 は、白黒ラインプリンタ (B&W LP) 101 と、カラーラインプリンタ (Color LP) 102 と、ハードディスク装置 (HDD) 103 と、スキャナ、ファクシミリ、メモリ、ネットワークインタフェースなどのハードウェアリソース 104 を有するとともに、プラットフォーム 120 と、アプリケーション 130 と、複合機初期化部 129 と、プログラム起動部 131 と、プログラミングサービス 132 とから構成されるソフトウェア群 110 とを備えている。

#### 【0043】

プラットフォーム 120 は、アプリケーションからの処理要求を解釈してハードウェア資源の獲得要求を発生させるコントロールサービスと、一又は複数のハードウェア資源の管理を行い、コントロールサービスからの獲得要求を調停するシステムリソースマネージャ（SRM）123 と、汎用 OS 121 とを有する。

#### 【0044】

コントロールサービスは、複数のサービスモジュールから形成され、SCS（システムコントロールサービス）122 と、ECS（エンジンコントロールサービス）124 と、MCS（メモリコントロールサービス）125 と、OCS（オペレーションパネルコントロールサービス）126 と、FCS（ファックスコントロールサービス）127 と、NCS（ネットワークコントロールサービス）128 とから構成される。また、プログラミングサービス 132 も、コントロールサービス層に含まれている。なお、このプラットフォーム 120 は、あらかじめ定義された関数により前記アプリケーション 130 から処理要求を受信可能とするアプリケーションプログラムインタフェース（API）を有する。

#### 【0045】

汎用 OS 121 は、UNIX（登録商標）などの汎用オペレーティングシステムであり、プラットフォーム 120 並びにアプリケーション 130 の各ソフトウェアをそれぞれプロセスとして並列実行する。

#### 【0046】

SRM 123 のプロセスは、SCS 122 とともにシステムの制御およびリソースの管理を行うものである。SRM 123 のプロセスは、スキャナ部やプリンタ部などのエンジン、メモリ、HDD ファイル、ホスト I/O（セントロ I/F、ネットワーク I/F、IEEE 1394 I/F、RS 232C I/F など）のハードウェア資源を利用する上位層からの要求にしたがって調停を行い、実行制御する。

#### 【0047】

具体的には、この SRM 123 は、要求されたハードウェア資源が利用可能であるか（他の要求により利用されていないかどうか）を判断し、利用可能であれば要求されたハードウェア資源が利用可能である旨を上位層に伝える。また、S

RM 1 2 3 は、上位層からの要求に対してハードウェア資源の利用スケジューリングを行い、要求内容（例えば、プリンタエンジンにより紙搬送と作像動作、メモリ確保、ファイル生成など）を直接実施している。

#### 【 0 0 4 8 】

SCS 1 2 2 のプロセスは、アプリ管理、操作部制御、システム画面表示、LED 表示、リソース管理、割り込みアプリ制御などを行う。

#### 【 0 0 4 9 】

ECS 1 2 4 のプロセスは、白黒ラインプリンタ (B&W LP) 1 0 1、カラーラインプリンタ (Color LP) 1 0 2、スキャナ、ファクシミリなどからなるハードウェアリソース 1 0 3 のエンジンの制御を行う。

#### 【 0 0 5 0 】

MCS 1 2 5 のプロセスは、画像メモリの取得および解放、ハードディスク装置 (HDD) の利用、画像データの圧縮および伸張などを行う。

#### 【 0 0 5 1 】

FCS 1 2 7 のプロセスは、システムコントローラの各アプリ層から PSTN / ISDN 網を利用したファクシミリ送受信、BKM (バックアップ SRAM) で管理されている各種ファクシミリデータの登録 / 引用、ファクシミリ読みとり、ファクシミリ受信印刷、融合送受信を行うための API を提供する。

#### 【 0 0 5 2 】

NCS 1 2 8 のプロセスは、ネットワーク I / O を必要とするアプリケーションに対して共通に利用できるサービスを提供するためのプロセスであり、ネットワーク側から各プロトコルによって受信したデータを各アプリケーションに振り分けたり、アプリケーションからデータをネットワーク側に送信する際の仲介を行う。具体的には、ftpd、lpd、snmpd、telnetd、smtpdなどのサーバデーモンや、同プロトコルのクライアント機能などを有している。

#### 【 0 0 5 3 】

OCS 1 2 6 のプロセスは、オペレータ (ユーザ) と本体制御間の情報伝達手段となるオペレーションパネル (操作パネル) 2 1 0 の制御を行う。OCS 1 2 6 は、オペレーションパネルからキー押下 (又はタッチ操作) をキーイベントと

して取得し、取得したキーに対応したキーイベント関数をSCS122に送信するOCSプロセスの部分と、アプリケーション130又はコントロールサービスからの要求によりオペレーションパネルに各種画面を描画出力する描画関数やその他オペレーションパネルに対する制御を行う関数などがあらかじめ登録されたOCSライブラリの部分とから構成される。このOCSライブラリは、アプリケーション130およびコントロールサービスの各モジュールにリンクされて実装されている。なお、OCS126のすべてをプロセスとして動作させるように構成しても良く、あるいはOCS126のすべてをOCSライブラリとして構成しても良い。

#### 【0054】

アプリケーション130は、ページ記述言語(PDL)、PCLおよびポストスクリプト(PS)を有するプリンタ用のアプリケーションであるプリンタアプリ111と、コピー用アプリケーションであるコピーアプリ112と、ファクシミリ用アプリケーションであるファックスアプリ113と、スキャナ用アプリケーションであるスキャナアプリ114と、ネットワークファイル用アプリケーションであるネットファイルアプリ115と、工程検査用アプリケーションである工程検査アプリ116とを有している。

#### 【0055】

アプリケーション130の各プロセス、コントロールサービスの各プロセスは、関数呼び出しとその戻り値送信およびメッセージの送受信によってプロセス間通信を行いながら、コピー、プリンタ、スキャナ、ファクシミリなどの画像形成処理にかかるユーザサービスを実現している。

#### 【0056】

このように、実施の形態1にかかる複合機100には、複数のアプリケーション130および複数のコントロールサービスが存在し、いずれもプロセスとして動作している。そして、これらの各プロセス内部には、一又は複数のスレッドが生成されて、スレッド単位の並列実行が行われる。そして、コントロールサービスがアプリケーション130に対し共通サービスを提供しており、このため、これらの多数のプロセスが並列動作、およびスレッドの並列動作を行って互いにプ



プロセス間通信を行って協調動作をしながら、コピー、プリンタ、スキャナ、ファクシミリなどの画像形成処理にかかるユーザサービスを提供するようになっている。

#### 【0057】

また、複合機100には、複合機100の顧客、サードベンダなどの第三者がコントロールサービス層の上のアプリケーション層に外部アプリを開発して搭載することが可能となっている。

#### 【0058】

なお、実施の形態1にかかる複合機100では、複数のアプリケーション130のプロセスと複数のコントロールサービスのプロセスとが動作しているが、アプリケーション130とコントロールサービスのプロセスがそれぞれ単一の構成とすることも可能である。また、各アプリケーション130は、アプリケーションごとに追加又は削除することができる。

#### 【0059】

また、複合機100はインストーラを含む。インストーラは、第三者が開発した外部アプリ117を、HDD103にインストールするものである。本実施の形態にかかる複合機100では、フラッシュカードなどのICカードの記憶媒体に外部アプリ117を格納し、インストーラによって外部アプリ117をHDD103にインストールし、プログラム起動部131によって、HDD103から外部アプリ117を起動してアプリケーション層で動作させるようになっている。一方、プリンタアプリ111、コピーアプリ112、ファックスアプリ113、スキャナアプリ114、ネットファイルアプリ115、工程検査アプリ116などの複合機100の出荷時に提供されるアプリケーション130、各コントロールサービス、HDD診断部132およびプログラム起動部131は、フラッシュメモリに出荷時に組み込まれており、複合機100の起動時（電源投入時）に複合機初期化部129によって起動されるようになっている。

#### 【0060】

図3に複合機100のハードウェア構成例を示す。

#### 【0061】

複合機 100 は、コントローラ 160 と、オペレーションパネル 175 と、ファックスコントロールユニット (FCU) 176 と、プリンタ等の画像形成処理に特有のハードウェア資源であるエンジン部 177 とを含む。コントローラ 160 は、CPU 161 と、システムメモリ 162 と、ノースブリッジ (NB) 163 と、サウスブリッジ (SB) 164 と、ASIC 166 と、ローカルメモリ 167 と、HDD 168 と、ネットワークインターフェースカード (NIC) 169 と、SDカード用スロット 170 と、USB デバイス 171 と、IEEE 1394 デバイス 172 と、セン트로ニクス 173 とを含む。なお、メモリ 162、167 は RAM、ROM 等を含む。FCU 176 およびエンジン部 177 は、コントローラ 160 の ASIC 166 に PCI バス 178 で接続されている。

#### 【0062】

CPU 161 が、複合機 100 にインストールされるアプリケーション、コントロールサービス等のプログラムを、メモリから読み出して実行する。

#### 【0063】

次に、以上のように構成された本実施の形態にかかる複合機 100 によるプログラム作成方法について説明する。まず、利用者は、PC 200 で WEB ブラウザを起動し、複合機 100 上にあるプログラミング画面の WEB ページの URL (例えば、http://www.xxx.yyy/zzz/debug.htm) を指定して複合機 100 のプログラミングサービス 132 にアクセスする。WEB ブラウザからのアクセスを受けたプログラミングサービス 132 は、プログラミング画面 201 (debug.htm) を PC 200 の WEB ブラウザに表示させる。

#### 【0064】

図 7 は、プログラミング画面 201 の内容の一例を示す説明図である。図 7 に示すように、プログラミング画面 201 には、プログラム入力フィールドと、参照ボタンと、アップロードボタンと、保存ボタンと、消去ボタンと、テスト実行ボタンと、実行設定ボタンが表示される。

#### 【0065】

ここで、プログラミング画面 201 は、html 形式 (Hyper Text

Markup Language) で記述されたファイル (debug.htm) がブラウザによって表示されたものである。また、プログラミングサービス 132 は、プログラミング画面 (debug.htm) からの操作入力や要求に応じたカスタマイズプログラムの入力に関する処理等を、CGI (Common Gateway Interface) のスクリプトを実行することによって行い、その処理結果をプログラミング画面 (debug.htm) に反映して、PC200 の WEB ブラウザに返す。

#### 【0066】

具体的には、プログラミングサービス 132 は、カスタマイズプログラムの入力処理等の要求 (ボタンのクリック操作等) があった場合には、debug.cgi のスクリプトを実行する。なお、本実施の形態では、CGI スクリプトを実行することにより、WEB アプリ 1117 による各処理を実行しているが、CGI スクリプトから呼び出される別のプログラムで実行するように構成しても良い。なお、プログラミング画面を、XML 形式 (eXtensible Markup Language) で記述するように構成することも可能である。

#### 【0067】

参照ボタンは、ブラウザが動作している PC200 の記憶領域に格納されたカスタマイズプログラムを参照するためのボタンであり、この参照ボタンを押下すると、PC200 の記憶領域内のディレクトリが表示され、所望のディレクトリに存在するカスタマイズプログラムを指定すると、ファイル名フィールドに指定されたカスタマイズプログラムのファイル名が表示される。なお、参照ボタンを押下したときに、複合機 100 の HDD 103 などの記憶媒体のディレクトリを参照するように構成しても良い。

#### 【0068】

アップロードボタンは、参照ボタンで指定したカスタマイズプログラムあるいは入力フィールドに入力したカスタマイズプログラムを、複合機 100 に転送 (アップロード) するためにクリック操作するものである。

#### 【0069】

プログラム入力フィールドは、カスタマイズプログラムを入力する領域である

。このプログラム入力フィールドには、PC200のキーボードなどの入力装置からカスタマイズプログラムを直接入力する他、上述のように参照ボタンをクリック操作することによってPC200の所望のディレクトリに存在する既存のプログラムファイルを指定することによってカスタマイズプログラムを入力する。`debug.htm`において、このプログラム入力フィールドに相当するすべての行の先頭には、「`program _list:`」の文字列が記述されている。

#### 【0070】

保存ボタンは、入力したカスタマイズプログラムを複合機100にアップロードした後に、カスタマイズディレクトリに格納する場合にクリック操作するものである。消去ボタンは、入力フィールドに入力されたカスタマイズプログラムを消去する際にクリック操作するものである。

#### 【0071】

実行設定ボタンは、カスタマイズディレクトリに保存されたカスタマイズプログラムを複合機100で実行できるように、カスタマイズプログラムとカスタマイズプログラムを実行させるボタンを対応づける際にクリック操作するものである。この実行設定ボタンが押下されると、プログラミングサービス132は、カスタマイズプログラム中で指定されたボタン（キー）を、カスタマイズディレクトリに存在する画面ファイル中のボタン（キー）からキーコードに基づいて検索し、カスタマイズプログラムとボタン（キー）との対応付けを行い、カスタマイズプログラムの名称とボタン（キー）のキーコードのキー対応付けテーブルを生成してHDD103又はフラッシュメモリなどに当該テーブルを格納する。

#### 【0072】

テスト実行ボタンは、入力されたカスタマイズプログラムを試験的に実行する際にクリック操作するものである。

#### 【0073】

図8は、入力するカスタマイズプログラムの内容の一例を示す説明図である。図8に示すように、本実施の形態では、カスタマイズプログラムはUNIX（登録商標）のシェルスクリプトである。入力されたシェルスクリプトは、汎用OS

121 (UNIX (登録商標)) のコマンド・インタプリタであるシェル 134 によってステップごとに逐次実行される。このため、入力したカスタマイズプログラムのコンパイルおよびリンクは不要である。

#### 【0074】

また、シェルスクリプトには、開発ライブラリで提供される複合機専用コマンドを指定できるようになっている。図 8 のスクリプトは、原稿をスキャナで読み込み (scanimage)、読み込み時に TIF 形式に圧縮された画像データを伸長し (decomp)、伸長した画像データに対し文字認識処理を行う (ocr)。そして、文字認識処理による認識結果をテキストデータとしてメール送信 (mail) する例を示している。このスクリプトの中で、scanimage, decomp, ocr の各コマンドが開発ライブラリで提供されるコマンドである。また、カスタマイズプログラムには、このシェルスクリプトを実行するためのボタン (キー) の指定が可能となっている。

#### 【0075】

PC200 の WEB ブラウザに表示されているプログラミング画面 201 (debug.htm) において、参照ボタンをクリックして PC200 に格納されているカスタマイズプログラムを指定し、あるいはプログラム入力フィールドに直接カスタマイズプログラムを入力し、アップロードボタンをクリック操作した場合、WEB ブラウザから複合機 100 のプログラミングサービス 132 に対しリクエストメッセージが送信される。このリクエストメッセージのメッセージボディには debug.htm が含まれており、この debug.htm には、アップロードボタンをクリック操作したときに debug.cgi を呼び出す旨が記述されている。このためリクエストメッセージを受信して解析を行ったプログラミングサービス 132 は、debug.cgi を実行する。

#### 【0076】

ここで、リクエストメッセージを受信した複合機 100 における処理について説明する。図 4 は、リクエストメッセージを受信した複合機 100 における処理の手順を示すフローチャートである。

#### 【0077】

h t t p d 1 0 6 は、ポート 80 番を常時監視しており、P C 2 0 0 からリクエストメッセージをポート 80 番で受信する（ステップ S 3 0 1）。そして、h t t p d 1 0 6 は、受信したリクエストメッセージを読み出し、リクエストメッセージを共有メモリ 1 0 5 に書き込む（ステップ S 3 0 2）。次に、h t t p d 1 0 6 は、リクエストメッセージ受信の旨を N C S 1 2 8 に通知する（ステップ S 3 0 3）。

#### 【0078】

通知を受けた N C S 1 2 8 は、さらにリクエストメッセージ受信の旨をプログラミングサービス 1 3 2 に通知する（ステップ S 3 0 4）。プログラミングサービス 1 3 2 は、リクエストメッセージ受信の旨の通知を受けたことをトリガとして、共有メモリ 1 0 5 を参照して、リクエストメッセージを読み出す（ステップ S 3 0 5）。そして、リクエストメッセージの内容を解析して（ステップ S 3 0 6）、リクエストメッセージのメッセージボディに記述された内容に応じた処理を実行する（ステップ S 3 0 7）。このとき、メッセージボディの h t m l 記述に C G I を実行する旨が記述されていれば、対応する C G I を実行する。

#### 【0079】

そして、プログラミングサービス 1 3 2 は、処理の実行結果をレスポンスメッセージとして、h t t p d 1 0 6 経由でリクエストメッセージの送信元である P C 2 0 0 に返信する（ステップ S 3 0 8）。

#### 【0080】

次に、プログラミングサービス 1 3 2 による d e b u g . c g i の処理について説明する。図 5 は、d e b u g . c g i の処理の手順を示すフローチャートである。なお、プログラミングサービス 1 3 2 が d e b u g . c g i のプログラムを含んでいてもよいし、d e b u g . c g i がプログラミングサービス 1 3 2 の外部に存在してもよい。以下、プログラミングサービス 1 3 2 が d e b u g . c g i のプログラムを含む場合について説明する。

#### 【0081】

プログラミングサービス 1 3 2 は、受信したリクエストメッセージの中の d e b u g . h t m を先頭から一行ずつ読み込む（ステップ S 4 0 1）。そして、読

み込んだ行が「program\_list:」の文字列を含むか否かを調べることにより、その行がプログラム入力フィールドか否かを判断する（ステップS402）。

#### 【0082】

そして、プログラム入力フィールドであると判断した場合には、プログラムファイルに書き出し（ステップS403）、読み込んだ一行をそのままPC200に送信する（ステップS404）。

#### 【0083】

そして、読み込んだ行が最終行か否かを判断し（ステップ405）、最終行であれば処理を終了し、最終行でなければ上記の処理を次の行に対して行う。これにより、複合機100において、プログラムファイルが作成され、PC200のWEBブラウザには、複合機100にアップロードされたプログラムの内容が順次表示されることになる。

#### 【0084】

上記の方法の他、PC200で入力したプログラムを複合機に送信するには種々の方法がある。例えば、アップロードボタンを押すことにより、PC200で入力したプログラムをPC200でプログラムファイルとし、それを複合機に送信する。複合機では、そのプログラムファイルの内容を格納し、必要に応じて格納したプログラムファイルの内容をPC200に送信する。

#### 【0085】

このようにプログラミング画面201から入力されたカスタマイズプログラムが複合機100にアップロードされ、プログラム入力フィールドにアップロード後のカスタマイズプログラムが表示された後、保存ボタン、実行設定ボタン、テスト実行ボタンのクリック操作が可能となる。これらのいずれかのボタンをクリック操作した場合におけるプログラミングサービス132の処理について説明する。

#### 【0086】

図6は、プログラミング画面201において、保存ボタン、実行設定ボタン、テスト実行ボタンのいずれかのクリック操作が行われた場合のプログラミングサ

ービス 132 の処理の手順を示すフローチャートである。以下の処理は、プログラミングサービス 132 における CGI スクリプトとしての J a v a (登録商標) S c r i p t が実行する。すなわち、プログラミング画面 201 におけるボタン操作を P C 200 が検出してそのイベントをプログラミングサービスに送信し、プログラミングサービス 132 はそのイベントに応じた J a v a (登録商標) S c r i p t を実行する。以下、J a v a (登録商標) S c r i p t がプログラミングサービス 132 に含まれているものとして説明する。

#### 【0087】

まず、プログラミングサービス 132 は、クリック操作されたボタンの種類を判断する(ステップ S 501)。プログラミング画面 201 で保存ボタンがクリックされた場合には、プログラミングサービス 132 は、ファイル名称を利用者に指定させ、カスタマイズプログラムを指定された名称で複合機 100 の HDD 103 のカスタマイズディレクトリに格納する(ステップ S 502)。

#### 【0088】

図 9 は、カスタマイズディレクトリに格納されたカスタマイズプログラムの名称の一例を示す説明図である。図 9 に示すように、複合機 100 の HDD 103 のカスタマイズディレクトリは、h d d / x x x / o p e p a n e となっており、このカスタマイズディレクトリにカスタマイズプログラムである s h e l l 1 , s h e l l 2 , s h e l l 3 が格納されている。

#### 【0089】

一方、プログラミング画面 201 で実行設定ボタンが押下された場合には、プログラミングサービス 132 は、上述のようにカスタマイズプログラムとボタンとの対応付けを行う(ステップ S 503)。

#### 【0090】

図 10 は、キー対応付けテーブルの内容の一例を示す説明図である。図 10 に示すように、キー対応付けテーブルには、ボタンのキーコードとカスタマイズプログラム名称が対応付けられている。

#### 【0091】

また、プログラミング画面 201 でテスト実行ボタンが押下された場合には、



プログラミングサービス 132 は、格納されたカスタマイズプログラムの実行を行う（ステップ S504）。これは、例えば、プログラミングサービス 132 がカスタマイズプログラムのシェルスクリプトを実行するためのコマンドを実行することによっておこなわれる。

#### 【0092】

このとき、テスト実行によって、オペレーションパネル 210 に表示される画面と同一の画面を PC200 で動作しているブラウザに表示する。これは、例えば、プログラミングサービス 132 が、プログラムのテスト実行により発生するオペレーションパネル画面への描画命令等を取得し、その描画命令に対応して表示される画面内容と同じ内容を PC200 の画面に表示するような命令を PC200 に対して送信することにより実現できる。

#### 【0093】

テスト実行によってエラーが出た場合には、PC200 でプログラムの修正を行い、修正したプログラムをアップロードし、再びテスト実行をする。

#### 【0094】

次に、このようにして生成されたカスタマイズプログラムの実行処理について説明する。図 11 は、複合機 100 の電源投入が行われてから、カスタマイズプログラムの実行が可能となるまでの一連の処理の手順を示すフローチャートである。

#### 【0095】

カスタマイズプログラムのカスタマイズディレクトリへの保存および起動設定ファイル 211 への設定が完了したら、利用者は複合機 100 の電源を再投入する。このとき複合機 100 では、ROM モニタ（図示せず）によってハードウェアの診断処理を行い（ステップ S1001）、その後汎用 OS 121 を起動し（ステップ S1002）、さらに複合機初期化部 129 を起動する（ステップ S1003）。複合機初期化部 129 は、まずコントロールサービスの起動を行い（ステップ S1004）、続いてコピーアプリ 112、プリンタアプリ 111 などの複合機 100 の出荷時に提供されている既存アプリを起動し（ステップ S1005）、さらに出荷後新たに開発された外部アプリなどのプログラムを起動

するためにプログラム起動部 131 を起動する（ステップ S1006）。

#### 【0096】

プログラム起動部 131 は、まず HDD 103 の起動設定ファイル 211 を参照し（ステップ S1007）、起動すべきプログラムの名称を取得する。

#### 【0097】

図 12 は、起動設定ファイル 211 の一例を示す説明図である。図 12 に示すように、起動設定ファイル 211 には、「プログラム名称 カスタマイズプログラム名」の形式で、プログラム起動部 131 によって起動されるプログラムが登録されている。図 12 の例では、シェルスクリプトであるシェルのプログラムが最初に登録され、その後に、外部アプリとして x x x アプリが登録されている。

#### 【0098】

このため、図 12 の例では、プログラム起動部 131 は、まずシェルを起動し、次いで、外部アプリである X X X アプリを順に起動するように設定されている。このため、プログラム起動部 131 は、まずインタプリタ 134（シェル）を起動し（ステップ S1008）、次いで、外部アプリである X X X アプリを順に起動する（ステップ S1009）。

#### 【0099】

オペレーションパネル 210 の操作表示部に表示されたボタン（キー）の中で、カスタマイズプログラムが割り付けられたボタン（キー）を押下（タッチ操作）すると、シェルスクリプトで記述されたカスタマイズプログラムがシェルによって実行される。これは例えば次のようにして実行される。

#### 【0100】

すなわち、カスタマイズプログラムが割り付けられたボタン（キー）の押下情報をプログラミングサービス 132 が取得し、キー対応付けテーブルを参照することによりそのボタンに対応するカスタマイズプログラムファイル名を取得する。そして、そのカスタマイズプログラムをシェルで実行するための命令を実行する。なお、上記の起動設定ファイルにカスタマイズプログラムを設定することにより、カスタマイズプログラムを実行してもよい。

#### 【0101】

このように実施の形態 1 にかかる複合機 100 では、プログラミングサービス 132 によって、ネットワーク上の PC 200 に、カスタマイズプログラムのプログラミング画面を表示し、この画面からカスタマイズプログラムを入力させ、複合機 100 のシェル 134 によって、カスタマイズプログラムを実行する。これにより、カスタマイズプログラムの開発において、プログラミング画面から追加、修正ができ、また追加修正による再コンパイル、再リンクなしに複合機 100 で実行でき、プログラムの作成を容易に行うことができる。

#### 【0102】

また、実施の形態 1 にかかる複合機 100 では、プログラミングサービス 132 によって、インターネットに接続された PC 200 に対しプログラミング画面を表示し、この画面からカスタマイズプログラムを入力させているので、カスタマイズプログラムの入力が容易な PC 200 で行うことができ、複合機 100 上でカスタマイズプログラムの作成編集を行う場合に比べて、カスタマイズプログラムの作成をより効率的に行うことができる。

#### 【0103】

なお、実施の形態 1 にかかる複合機 100 では、インターネット 220 に接続されたクライアント端末としての PC 200 の要求により、カスタマイズプログラムの入力処理を行っているが、インターネット以外のネットワーク、例えば LAN などに接続されたクライアント端末からカスタマイズプログラムの入力を行うように構成することも可能である。また、通信プロトコルとしては、TCP/IP 以外のプロトコルを利用しても良い。

#### 【0104】

また、上記の実施の形態では、複合機から PC 200 に対してプログラム入力画面を送信しているが、PC 200 が本実施の形態におけるプログラムの入力、送信などを行うプログラムを備えていてもよい。

#### 【0105】

##### (実施の形態 2)

実施の形態 1 にかかる複合機 100 は、ネットワーク接続されたクライアント端末である PC 200 でカスタマイズプログラムを作成して複合機 100 に保存

するものであったが、この実施の形態 2 にかかる複合機は、カスタマイズプログラムの入力を複合機 100 のオペレーションパネルから行うものである。

#### 【0106】

図 13 は、実施の形態 2 にかかる複合機 1100 における主要構成を示すブロック図である。なお、複合機 1100 の他の構成については、図 2 に示した実施の形態 1 の複合機 100 の構成と同様である。図 13 に示すように、本実施の形態にかかる複合機 1100 は、複合機初期化部 129 と、プログラム起動部 131 と、プログラミングサービス 1132 と、インタプリタ（シェル） 134 と、HDD 103 とを備えた構成となっている。

#### 【0107】

プログラミングサービス 1132 は、オペレーションパネル 210 の操作表示部にプログラミング画面を表示し、表示されたプログラミング画面からカスタマイズプログラムを入力させるものである。また、プログラミングサービス 1132 は、入力されたカスタマイズプログラムを HDD 103 のカスタマイズ領域として作成されたカスタマイズディレクトリ 212 に保存する。さらに、プログラミングサービス 1132 は、複合機 1100 上でカスタマイズプログラムが実行されるように、起動設定ファイルに入力されたカスタマイズプログラムを登録することもできる。

#### 【0108】

なお、複合機初期化部 129、プログラム起動部 131、シェル 134 の機能および HDD 103 の起動設定ファイル 211 およびカスタマイズ領域としてのカスタマイズディレクトリについては、実施の形態 1 にかかる複合機 100 と同様である。

#### 【0109】

図 14 は、プログラミング画面 1101 の一例を示す説明図である。本実施の形態にかかる複合機 1100 において、カスタマイズプログラムを作成するためには、例えば、システム初期設定画面からプログラミングのタグを選択してプログラミング画面を表示させる。

#### 【0110】

このプログラミング画面 1101 は、実施の形態 1 における同画面とはほぼ同様に、入力フィールドと、参照ボタンと、保存ボタンと、実行設定ボタンと、テスト実行ボタンとが表示されている。かかるプログラミング画面 1101 では、タッチ操作でキー入力を行うソフトウェアキーボードを操作表示部に表示させて、入力フィールドにカスタマイズプログラムを入力することができる。

#### 【0111】

しかしながら、操作表示部の領域サイズや入力の困難さから直接カスタマイズプログラムを入力することは困難である。このため、予め HDD 103 などの記憶媒体の所望の領域に、PC などで作成したカスタマイズプログラムをネットワーク経由で格納し、プログラミング画面 1101 上の参照ボタンを押下して、カスタマイズプログラムを選択することにより、入力フィールドにカスタマイズプログラムを表示し、ソフトウェアキーボードを表示させ、プログラムの簡単な修正などだけを入力フィールドで行うことが好ましい。

#### 【0112】

そして、カスタマイズプログラムが完成したら、保存ボタンを押下する。これによって、実施の形態 1 と同様に、プログラミングサービス 1132 によって、カスタマイズプログラムを HDD 103 のカスタマイズディレクトリに保存する。また、実行設定ボタンを押下すると、実施の形態 1 と同様に、プログラミングサービス 1132 によって、カスタマイズプログラムとカスタマイズプログラムで指定されたボタン（キー）のキーコードが対応付けられてテーブルとして保存される。なお、作成したカスタマイズプログラムの実行処理については、実施の形態 1 の複合機 100 と同様である。

#### 【0113】

このように実施の形態 2 にかかる複合機 1100 では、プログラミングサービス 1132 によって、オペレーションパネル 210 の操作表示部にプログラミング画面 1101 を表示し、この画面からカスタマイズプログラムを入力させているので、ネットワークに接続されていないスタンドアローンの複合機 1100 においてもカスタマイズプログラムを作成することができる。

（実施の形態 3）

実施の形態 1 および 2 の複合機 100, 1100 は、カスタマイズプログラムと操作表示部のボタン（キー）との対応付けをカスタマイズプログラムの記述により行うものであったが、この実施の形態 3 にかかる複合機は、複合機の電源投入時にカスタマイズプログラムと操作表示部のボタン（キー）との対応付けを動的に行うものである。

#### 【0114】

図 15 は、実施の形態 3 にかかる複合機 1300 の機能的構成を示すブロック図である。本実施の形態の複合機 1300 では、実施の形態 1 と同様に、ネットワーク上の PC 200 でカスタマイズプログラムの作成を行って、複合機 1300 の HD 103 に作成されたカスタマイズプログラムを保存して実行する。

#### 【0115】

図 15 に示すように、本実施の形態の複合機 1300 は、オペパネ登録部 133 を備えている点が図 2 に示す実施の形態 1 の複合機 100 と異なっており、他の構成は実施の形態 1 の複合機 100 と同様である。

#### 【0116】

オペパネ登録部 133 は、カスタマイズプログラムを実行させるためのボタン（キー）の割り付けを行うものである。すなわち、オペパネ登録部 133 によって割り付けたボタン（キー）をオペレーションパネル上で押下（タッチ操作）することによって、カスタマイズプログラムが実行される。

#### 【0117】

PC 200 上からのカスタマイズプログラムの作成および保存の処理は、実施の形態 1 のプログラミングサービス 131 と同様に行われる。

#### 【0118】

以下、カスタマイズプログラムと操作表示部に表示されるボタン（キー）の割付処理について説明する。カスタマイズプログラムのカスタマイズディレクトリへの保存が完了すると、作成したカスタマイズプログラムをオペレーションパネル 210 のボタン（キー）に割り付けるために、利用者は複合機 100 の電源を再投入する。

#### 【0119】

図16は、複合機1300の電源投入が行われてから、カスタマイズプログラムとボタン（キー）との割り付けが完了するまでの一連の処理の手順を示すフローチャートである。

#### 【0120】

複合機1300の電源投入からプログラム起動部131までの処理（ステップS1501～S1506）については、実施の形態1の複合機100による処理（ステップS1001～S1006）と同様である。

#### 【0121】

起動されたプログラム起動部131は、まずHDD103の起動設定ファイル211を参照し（ステップS1507）、起動すべきプログラムの名称を取得する。図17は、本実施の形態で使用される起動設定ファイル211の内容の一例を示す説明図である。この例では、オペパネ登録部のプログラム、シェルおよび外部アプリであるXXX アプリを順に起動するように設定されている。

#### 【0122】

このため、プログラム起動部131は、まずオペパネ登録部133のプログラムを起動する（ステップS1508）。なお、このオペパネ登録部133のプログラムは、予めカスタマイズディレクトリに格納されている。次いで、インタプリタ（シェル）134を起動し（ステップS1509）、最後に、登録されている外部アプリを起動する（ステップS1510）。

#### 【0123】

起動されたオペパネ登録部133は、キー割り付け設定画面をオペレーションパネルに表示する（ステップS1511）。図18は、キー割り付け設定画面の一例を示す説明図である。図18に示すように、キー割り付け設定画面には、各カスタマイズプログラムごとに割り付け可能なボタン（キー）が表示され、かかる画面で所望のボタン（キー）を押下（タッチ操作）すると、オペパネ登録部133は、該当するカスタマイズプログラムの名称と押下されたボタン（キー）のキーコードとを対応付けて、HDD103又はフラッシュメモリなどの記憶媒体にテーブル情報として格納する（ステップS1512）。これによって、オペレーションパネルに表示されるボタン（キー）にカスタマイズプログラムが割り付

けられることになる。

#### 【0124】

このように実施の形態3にかかる複合機1300では、プログラミングサービス132によって、プログラミング画面201を表示してカスタマイズプログラムを入力させ、オペパネ登録部133によって、カスタマイズプログラムを操作表示部に表示および入力を行うボタン（キー）に割り付ける。このオペパネ登録部133を複合機1300の電源投入時に起動することにより、カスタマイズプログラムが実行される場合のボタン操作を複合機1300の起動時に動的に定めることができる。

#### 【0125】

なお、本実施の形態では、オペパネ登録部133によってボタン（キー）への割り付け処理を行っているが、例えば、SCS122などのコントロールサービスでボタン（キー）割り付け処理を行うようにしてもよい。これにより、割り付け処理のために別途プログラムを起動する必要がなくなり、効率的な処理が行える。

#### 【0126】

##### （実施の形態4）

実施の形態1～3の複合機100，1100，1300は、シェルスクリプトとしてカスタマイズプログラムを複合機100，1300の汎用OS121で用意されたコマンドインタプリタであるシェル134によって実行するものであったが、この実施の形態4にかかる複合機は、シェル134以外のインタプリタを使用してカスタマイズプログラムを実行するものである。

#### 【0127】

図19は、実施の形態4にかかる複合機1700を備えたネットワーク構成図であり、図20は、実施の形態4にかかる複合機1700の機能的構成を示すブロック図である。実施の形態4にかかる複合機1700は、実施の形態1と同様に、TCP/IPを利用したインターネットに接続され、当該ネットワークに接続されたクライアント端末であるPC200からカスタマイズプログラムの入力およびデバッグ作業を行うことを可能としたものである。なお、カスタマイズプ



プログラムとしては、例えば、オペレーションパネル 210 の操作表示部にボタンを表示し、あるボタンを押したときにそのボタンに対応した処理を行うためのプログラムがある。

#### 【0128】

本実施の形態の複合機 1700 では、カスタマイズプログラムを、Visual Basic（登録商標）言語で記述する。そして、作成されたカスタマイズプログラムを、汎用 OS 121 の配下で動作する Visual Basic インタプリタで実行させている。

#### 【0129】

図 19 に示すように、複合機 1700 は、複合機初期化部 129 と、プログラム起動部 131 と、プログラミングサービス 132 と、Visual Basic インタプリタ 1734（以下、「VB インタプリタ 1734」という。）と、NCS 128 と、httpd 106 と、共有メモリ 105 と、HDD 103 とを備えた構成となっている。なお、プログラミングサービス 132、複合機初期化部 129、プログラム起動部 131、NCS 128、httpd 106 の構成および機能、HDD 103 に格納される起動設定ファイル 211 の内容およびカスタマイズディレクトリについては、実施の形態 1 の複合機 100 と同様である。

#### 【0130】

VB インタプリタ 1734 は、カスタマイズプログラムをステップごとに解析し逐次実行するものである。本実施の形態において、カスタマイズプログラムの作成および保存は、実施の形態 1 のプログラミングサービス 132 と同様に行われる。

#### 【0131】

また、本実施の形態の複合機 1700 では、VB インタプリタ 1734 によってカスタマイズプログラムを実行させるため、複合機 1700 の起動時に VB インタプリタ 1734 も起動させておく必要がある。このため、実施の形態 1 と同様の起動設定ファイル 211 に VB インタプリタ 1734 のプログラム名称を予め設定しておく。

#### 【0132】

図 2 1 は、起動設定ファイルの内容の一例を示す説明図である。図 2 1 に示すように、本実施の形態の複合機 1 7 0 0 で使用する起動設定ファイル 2 1 1 には、先頭の行に「VB インタプリタプログラム」が設定されている。このため、プログラム起動部 1 3 1 は、起動設定ファイル 2 1 1 を参照した後、まず VB インタプリタ 1 7 3 4 を起動することになり、これによってカスタマイズプログラムの実行が可能となる。

#### 【0133】

このように実施の形態 4 にかかる複合機 1 7 0 0 では、汎用 OS 1 2 1 の配下で動作する VB インタプリタによってカスタマイズプログラムを実行するので、汎用 OS で提供されるインタプリタよりも高度な処理を行えるインタプリタを利用することができる。従って、UNIX（登録商標）のシェルを用いるよりも高度なプログラムを容易に作成することが可能となる。

#### 【0134】

なお、実施の形態 4 では、PC 2 0 0 上では VB インタプリタはインストールされていないが、PC 2 0 0 に Visual Basic 開発環境をインストールし、操作表示部の画面設計を含むプログラム作成を PC 2 0 0 上で行って、生成されたプログラムをファイル転送などを利用して、複合機 1 7 0 0 のカスタマイズディレクトリに格納するように構成しても良い。

#### 【0135】

また、実施の形態 4 の複合機 1 7 0 0 では、インタプリタとして VB インタプリタ 1 7 3 4 を使用し、開発するカスタマイズプログラムを Visual Basic 言語で記述していたが、この他、インタプリタ用の言語を用いても良い。例えば、複合機 1 7 0 0 に Java（登録商標）VM（仮想マシン）を搭載し、カスタマイズプログラムを Java（登録商標）言語で記述するように構成することもできる。

#### 【0136】

実施の形態 1 ～ 4 にかかる複合機では、プログラミングサービス 1 3 2 をコントロールサービス層に搭載した構成としているが、アプリケーション層に搭載した構成とすることもできる。

## 【0 1 3 7】

## (実施の形態 5)

実施の形態 4 にかかる複合機は、汎用 OS 1 2 1 の配下で動作するインタプリタを使用してカスタマイズプログラムを実行するものであったが、この実施の形態 5 にかかる複合機 1 8 0 0 は、VB インタプリタ 1 8 0 1 と VB アプリ 1 8 0 2 をアプリケーション層に搭載した構成としたものである。

## 【0 1 3 8】

図 2 2 に、実施の形態 5 に係る複合機の機能構成のブロック図を示す。実施の形態 5 にかかる複合機は、実施の形態 4 と同様に、TCP/IP を利用したインターネットに接続され、当該ネットワークに接続されたクライアント端末である PC 2 0 0 から、カスタマイズプログラムの入力およびデバッグ作業を行うことを可能としたものである。ただし、実施の形態 5 では、作成されたカスタマイズプログラム（以下、VB アプリと称する）を、アプリケーション層に搭載された Visual Basic インタプリタ（以下、VB インタプリタと称する）で実行させている。なお、図中、プログラミングサービス 1 3 2 をアプリケーション層に搭載した構成としてもよい。

## 【0 1 3 9】

Visual Basic 開発環境をインストールした PC 2 0 0 上で VB アプリの作成を行い、その作成した VB アプリを複合機に転送し、複合機でその VB アプリを格納しておき、オペレーションパネルにおけるアプリ選択等に応じて VB アプリを起動して VB インタプリタ 1 8 0 1 により実行するようにする。

## 【0 1 4 0】

以下、より具体的に説明する。

## 【0 1 4 1】

図 2 3 は、PC 2 0 0 上で Visual Basic 開発環境を用いて VB アプリを作成している場面の例を示す図である。

## 【0 1 4 2】

同図中の画面内には Project 2 -Form 2 (Form) の表題がついたウィンドウと、Project 2 -Form 2 (コード) の表題がついたウィンドウが示されている。

**【 0 1 4 3 】**

Project 2 -Form 2 (Form) の表題がついたウィンドウは、複合機のオペレーションパネルで表示されるメニュー画面を作成するためのものであり、表示されている状態では、3つのボタンが設定されており、一つは"SCAN"、一つは"FAX"あと一つは"START"のボタンである。これらのボタンが押された場合に、複合機のスキニングやファックス送信などの動作が行われるように設計される。

**【 0 1 4 4 】**

Project 2 -Form 2 (コード) の表題がついたウィンドウは、上記のパネルの画面中にあるボタンが押されたときの動作を記述するプログラミングを行うためのものである。

**【 0 1 4 5 】**

図 2 3 に示す状態でプログラムの保存を行うと、Project 2 .vbp と Form 2 .frm という二つのファイルが生成される。Project 2 .vbp にはプロジェクトを管理するための情報や、プログラムを実行する際の条件となるものが、変数への値の設定として保存されている。

**【 0 1 4 6 】**

Form 2 .frm は、VB インタプリタ 1 8 0 1 によりインタプリトされるプログラムそのものであり、この VB アプリを PC 2 0 0 から複合機上のプログラミングサービス 1 3 2 を介してアップロードする。

**【 0 1 4 7 】**

そして、プログラミング画面 2 0 1 でテスト実行ボタンが押下された場合には、複合機上の VB インタプリタ 1 8 0 1 が起動され、入力された VB アプリの実行が行われる。このとき、テスト実行によって、オペレーションパネル 2 1 0 に表示される画面と同一の画面を PC 2 0 0 で動作しているブラウザに表示する。

**【 0 1 4 8 】**

図 2 4、2 5 に、VB インタプリタ 1 8 0 1 により実行される VB アプリの例を示す。

**【 0 1 4 9 】**

図 2 4 に示す VB プログラムが複合機において実行されると、まず、NEXT

WIN ボタンを含むウィンドウが表示される。そして、NEXTWIN ボタンが押されると、call("change\_display", "Form2")により、関数change\_display("Form2")が実行され、次の画面 (Form2 ウィンドウ) に切り替わる。

#### 【0150】

図 25 に、そのForm2 ウィンドウを表示するVB プログラムを示す。なお、Form2 のVB プログラムは、図 23 に示した画面に対応する。

#### 【0151】

```
Form2 ウィンドウでは、SCAN ボタンが押されたときに、  
call("ScanStart", A4, ADF, TIFF, BINARY, "/work/tmpfile.tif")  
gwOpItemCreate(win, MESSAGE_ITEM,  
                ITEM_MESSAGE_X,      0,  
                ITEM_MESSAGE_Y,      12,  
                ITEM_MESSAGE_WIDTH,   100,  
                ITEM_MESSAGE_HEIGHT,  12,  
                ITEM_MESSAGE_CENTER_X, 0,  
                ITEM_MESSAGE_BLINK,   0,  
                ITEM_MESSAGE,         "文字列", 0,  
                ITEM_MESSAGE_FONT,    FONT_12, 0,  
                0);
```

が実行される。

#### 【0152】

最初のScanStart関数は、A4サイズ of 原稿をADFから読み込んで、2 値画像としてTIFFフォーマットでHDD上のファイル"/work/tempfile"で保管することを意味している。この関数をVB インタプリタ1801が解釈して、複合機に動作を行わせることとなる。

次の行のgwOpItemCreate()は複合機におけるOCS126によりその機能が提供されている関数であり、現在表示されている画面に文字列を表示する機能を実行するものである。gwOpItemCreate()は、VB インタプリタ1801により、OCS126 への関数コールとして実行される。

**【0153】**

このようにして作成されたVBアプリを複合機で実行させるには、図11に示した起動設定ファイルにVBインタプリタとVBアプリを設定し、例えば図10に示した手順と同様にして、VBインタプリタとVBアプリを起動する。

**【0154】**

複合機でのVBアプリ実行時、オペレーションパネルに表示操作を行うためには次のような処理を行う。まず、VBインタプリタ1801は、SCS122からの画面描画準備完了イベントを待ち、上記のイベントを受け取ったら、初期表示するウィンドウイメージを作成する（ルートウィンドウ生成）。次に、VBアプリが、ルートウィンドウ内の子ウィンドウの生成や文字列表示の関数を実行することにより初期画面を準備する。そして、画面準備完了を示す関数を実行することにより、画面の準備ができたことをSCS122に通知する。次に、オペレーションパネル上でアプリ選択ボタンが選択されると、SCS122から操作部オーナー移行要求のイベントが発行される。このイベントをVBアプリが受け取ると、操作部オーナー移行可能であることを示す関数を実行し、この時点で上記の初期画面が表示される。この後、VBアプリの実質的な処理がVBインタプリタ1801により実行される。

**【0155】**

上記のように、本実施の形態によれば、VBインタプリタを用いることにより効率良くアプリを開発することが可能となる。

**（実施の形態6）**

実施の形態5にかかる複合機は、VBインタプリタをアプリケーション層に搭載した構成としたものであったが、実施の形態6にかかる複合機は、Java（登録商標）実行環境をアプリケーション層に搭載した構成としたものである。

**【0156】**

図26に、実施の形態6に係る複合機1900の機能構成のブロック図を示す。同図に示すように、実施の形態6に係る複合機1900は、アプリケーション層にJava（登録商標）実行環境1901とJava（登録商標）アプリ1902を搭載した構成を有している。Java（登録商標）アプリケーションをP

C (パーソナルコンピュータ) などで開発し、それを複合機にダウンロードして J a v a (登録商標) 実行環境 1901 を用いて実行する。なお、この P C は、C P U、メモリ、ハードディスクなどを有する一般的なコンピュータである。

#### 【0157】

図 27 は、P C における J a v a (登録商標) 開発環境と複合機 1900 における J a v a (登録商標) 実行環境の構成を示す図である。

#### 【0158】

同図に示すように、複合機 1900 における J a v a (登録商標) 実行環境は、クラスライブラリ 1911、仮想マシン 1912、アプリケーション管理部 1914 を有している。

#### 【0159】

クラスライブラリ 1911 は、J a v a (登録商標) アプリが容易に複合機を操作するためのサービスを提供するために用いられるクラスライブラリである。本実施の形態におけるクラスライブラリは、例えば、操作パネルクラス、イベントクラス、複合機制御クラス、ネットワークトランザクションクラス、基本的な J a v a (登録商標) クラスライブラリ等から構成される。なお、J a v a (登録商標) アプリ自身もクラスの集合であり、そのクラスがクラスライブラリのクラスを呼び出すことにより、アプリの処理が実行される。J a v a (登録商標) アプリに、それが使用するクラスライブラリのクラスの機能を全て含めることにより、クラスライブラリ 1911 を搭載しない構成もあり得る。

#### 【0160】

J a v a (登録商標) のソースコードプログラムはコンパイルして中間コード形式のバイトコードされ、複合機にロードされる。仮想マシン 1912 はそのバイトコードを解釈・実行する。

#### 【0161】

アプリケーション管理部 1914 は、J a v a (登録商標) アプリを管理する機能を有しており、例えば、J a v a (登録商標) アプリのリスト表示、起動や強制終了などの J a v a (登録商標) アプリ実行管理、J a v a (登録商標) アプリのロードやバージョンアップ、インストール済みの J a v a (登録商標) ア

プリの削除、アプリケーション登録のためのパスワード設定等を行う機能を有している。

#### 【0162】

より詳細には、図28に示すように、J a v a（登録商標）の実行環境は更にネイティブプログラムインタフェース1913を有している。ネイティブプログラムインタフェース1913は、仮想マシン1912で実行されるJ a v a（登録商標）コードが、C言語などの他のプログラミング言語で書かれたアプリケーションやライブラリと相互運用するための機能を有している。このネイティブプログラムインタフェース1913の機構を用いて、アプリはコントロールサービスのA P Iにアクセスできる。

#### 【0163】

図27に示すJ a v a（登録商標）開発環境は、J a v a（登録商標）の実行環境と同様のクラスライブラリ1921、仮想マシン1922、アプリケーション管理部1924を有している。また、更に、バイトコードを生成するためのJ a v a（登録商標）コンパイラ1925と、アプリによる複合機の操作をエミュレートするエミュレータ1923を有している。このJ a v a（登録商標）開発環境で開発されたJ a v a（登録商標）アプリは、I Cカード等の記録媒体を介して複合機のJ a v a（登録商標）実行環境にロードできる。また、ネットワークを介してロードすることもできる。

#### 【0164】

例えば、図29に示すように、W e bサーバにJ a v a（登録商標）開発環境で開発したJ a v a（登録商標）アプリを格納しておき、J a v a（登録商標）実行環境からW e bサーバにアクセスしてJ a v a（登録商標）アプリをロードするように構成できる。また、W e bサーバの代わりにF T Pサーバを使用することもできる。

#### 【0165】

（J a v a（登録商標）アプリケーションの開発）

次に、J a v a（登録商標）開発環境でのJ a v a（登録商標）アプリの開発について詳細に説明する。



## 【0166】

図30は、Java（登録商標）アプリ開発の流れを示す図である。このように、まず、Java（登録商標）のソースコードをJava（登録商標）ファイルとして作成し、これをJava（登録商標）コンパイラでコンパイルし、クラスファイルとする。そして、複数のクラスをまとめてJarファイルとする。なお、Jarファイルにはmainルーチンを持つクラスを示すファイルを含めておくことが好ましい。

## 【0167】

このようにして作成したJava（登録商標）アプリを、これまでに説明した実施の形態と同様にして複合機上でテスト実行することにより、デバッグを行うことが可能であるが、本実施の形態では、図27に示すエミュレータ1923を用いることにより、ターゲットである複合機を用いずにJava（登録商標）アプリのデバッグをすることが可能である。これは、Java（登録商標）では、仮想マシンを使用する構成となっているため、実行環境をエミュレートすることが比較的容易であることによる。エミュレータを用いた開発が完了したら、Java（登録商標）アプリ（バイトコード）をネットワーク又はICカードを介して複合機にロードすることにより、複合機を用いた評価を行うことが可能である。

## 【0168】

エミュレータは、Java（登録商標）アプリ側からの命令を受信し、それに応じた動作を行うように記述したプログラムとして構成することができる。例えば、複合機のオペレーションパネルの表示部への表示命令に対しては、複合機上に表示する画面と同様の画面をPCのディスプレイ上に表示する。ただし、開発環境であるPCには複合機におけるエンジン（スキャナ、プリンタなど）がないので、例えば、Java（登録商標）アプリからのプリント命令に対しては、PCの画面上にプリントを行う画像を表示したり、PC自身に接続されているデフォルトのプリンタに印刷を行う。また、ユーザがコピー操作をPCのエミュレータ画面上でした場合には、例えば、ユーザにコピー対象となるファイルをローカルファイルシステムから選ぶように通知し、コピーのあて先を尋ねるダイアログ

ボックスを表示し、選択されたファイルをその指定された宛先にコピーする。

#### 【0169】

なお、エミュレータは、図27に示すように仮想マシンとクラスライブラリとは別のプラグラムとして実装することもできるし、図31に示すように、クラスライブラリの中のクラスとして実装することもできる。

#### 【0170】

図32に、エミュレータによりPCに表示されるオペレーションパネル画面を示す。この画面に表示されるボタンなどの要素は、実際の複合機のものと同等である。また、エミュレータは、格納したJava（登録商標）アプリを選択して実行するローダーの機能も有している。ローダーとして動作する場合には、例えば、図33に示す画面を表示する。そして、ユーザはJava（登録商標）アプリを選択し、実行ボタンを押すことによりJava（登録商標）アプリが実行される。なお、ここで表示されるJava（登録商標）アプリは、mainルーチンを持つクラス名、もしくはJarファイル名である。

#### 【0171】

エミュレータは、エミュレータ上で動作するアプリケーションの状態状態情報を自動的にログファイルに収集する機能も有しており、どのようなメッセージをログとして残すかの設定もできる。例えば、アプリを異常終了させるような深刻なエラーのみをログとして収集するように設定したり、アプリケーションのデバッグに有益な詳細情報をログとして収集するように設定することができる。

#### 【0172】

エミュレータを用いて開発環境側でプログラムのデバッグを行うことは、他の実施の形態においても適用可能である。すなわち、例えば実施の形態1において、PC200にインタプリタと、本実施の形態と同様のエミュレータを搭載する。プログラムのデバッグが進んだところで、評価のために複合機にアクセスしてプログラムを送信する。

#### 【0173】

（Java（登録商標）アプリケーションの例）

次に、Java（登録商標）アプリケーションの例について説明する。ここで

は、操作パネルクラスを使用したサンプルプログラムを使用した J a v a（登録商標）アプリの例について説明する。図 3 4 に、操作パネルクラスの階層構成を示す。

#### 【 0 1 7 4 】

図 3 4 中、\*\*\* W i n d o w のクラスはウィンドウ作成のためのクラスであり、\*\*\*\* I t e m のクラスは、ボタンなどのウィンドウ中におけるエレメントを作成するためのクラスである。例えば、以下に示すコードを含むプログラムにより、図 3 5 に示すようなメインウィンドウとその中のエレメントが作成される。

#### 【 0 1 7 5 】

```
mainwindow = new PanelWindow (panel.root(), 640, 240); (a)
```

```
MessageItem msg = new MessageItem (20, 20, 600, 40); (b)
```

```
ButtonItem Button = new ButtonItem (10, 60, 80, 30); (c)
```

上記の (a) は、幅 6 4 0 ドット、高さ 2 4 0 ドットのメインウィンドウを作成するためのコードである。(b) は、メインウィンドウの位置 (2 0、2 0) に幅 6 0 0、高さ 4 0 のメッセージアイテムを作成するためのコードである。(c) は、(1 0、6 0) の位置に、サイズ 8 0 × 3 0 のボタンを作成するためのコードである。

#### 【 0 1 7 6 】

図 3 6 に示すプログラムは、複合機のオペレーションパネルにユーザーインタフェースを実装するプログラムである。図 3 7 に示すように、このプログラムは起動直後に” Hello World” 及び” Your Input Was…” のメッセージと” Get Input…” のボタンを表示する。ユーザーが” Get Input…” のボタンに触れると、パネル上にソフトキーボードを表示し、ソフトキーボードには、” Add your Input” のタイトルが表示され、ユーザーに任意のコード入力を促す。ユーザーが、ソフトキーボードから任意の文字列、例えば、” ABCD!” を入力した場合、パネル上の” Your Input Was…” のメッセージの下にユーザーが入力した文字列を表示、すなわち、” ABCD!” を表示する。

#### 【 0 1 7 7 】

以下、図 36 に示すサンプルプログラムの内容について、プログラム中に記述した注釈の番号に沿って説明する。

#### 【0178】

①は、操作パネルクラスライブラリを使用するための命令である。②は、GWApp を継承することを示し、これにより複合機のアプリケーション雛型（抽象）を使用することが可能となる。また、複雑な初期設定や終了処理をユーザーが記述せずに済み、メッセージ受信などの処理もユーザーに対して隠蔽することができる。また、③により“Hello World”のメッセージを表示するオブジェクトを生成し、④により“Get Input...”のボタンオブジェクトを生成する。そして、⑤により“Your Input Was...”のメッセージを表示するオブジェクトを生成し、⑥によりソフトキーボードから入力した文字を表示するメッセージオブジェクトを生成する。

#### 【0179】

(Java (登録商標) 実行環境における Java (登録商標) アプリの実行について)

図 27 に示す構成において、アプリケーション管理部 1914 の機能を用いることにより、以下のような手順でダウンロードから起動までの処理を行うことが可能である。

#### 【0180】

アプリケーション管理部 1914 は、他のアプリケーションと同様にして複合機の起動時に起動され、アプリケーション起動キー等を押すことにより、オペレーションパネルにユーザーインタフェースを表示する。このとき、Java (登録商標) アプリがロードされていない場合は、アプリケーションロード画面を表示する。ユーザーは、このアプリケーションロード画面から所定の Web サイト又は、IC カードにアクセスし、Java (登録商標) アプリをロードする。

#### 【0181】

ロードの処理においては、Java (登録商標) アプリをロードする前に、当該 Web サーバ等に対して Java (登録商標) アプリに関する情報の問い合わせを行い、インストール可能である Java (登録商標) アプリであるか否かを

確認する。確認項目は、例えば、プログラムサイズ、バージョン確認、プログラム最終更新情報、使用するメモリワークサイズ、使用するストレージサイズ、あらかじめ使用するネットワークアドレスの確認、利用可能期間（利用回数）、アプリケーションプログラム名等である。

#### 【0182】

上記の事項を確認し、インストール可能であると判断した場合は、アプリケーション管理部 1 9 1 4 が J a v a（登録商標）アプリケーションファイルを複合機にダウンロードする。

#### 【0183】

そして、J a v a（登録商標）アプリケーションプログラムのダウンロードが終了すると、アプリケーション管理部 1 9 1 4 は、アプリケーション名を取得し、アプリケーションリストに追加し、同時に J a v a（登録商標）アプリケーションプログラムを複合機のハードディスクに格納する。

#### 【0184】

以上の処理が完了することにより、ダウンロードした J a v a（登録商標）アプリを利用できる状態になり、ユーザーは、アプリケーション管理部 1 9 1 4 が提供する J a v a（登録商標）アプリケーション起動画面から起動する J a v a（登録商標）アプリを選択することにより、仮想マシンを起動し、J a v a（登録商標）アプリを実行することができる。

#### 【0185】

次のようにして J a v a（登録商標）アプリを実行することも可能である。次に説明する方法は、図 2 9 に示した構成において、J a v a（登録商標）アプリを W e b サーバからロードして実行する例である。以下の処理は、アプリケーション管理部 1 9 1 4 に含まれるローダー（プログラム）により実行される。なお、ローダー自身は J a v a（登録商標）アプリとして実装してもよいし、例えば C 言語で実装してもよい。

#### 【0186】

図 3 8 に、図 2 9 に示した構成における J a v a（登録商標）アプリのアップロードからダウンロードまでの概念図を示す。このように、J a v a（登録商標

）開発環境において、開発、テストされた J a v a（登録商標）アプリを W e b サーバにアップロードする（ステップ 1）。複合機のローダーは、W e b サーバにアクセスし（ステップ 2）、ユーザ所望の J a v a（登録商標）アプリをダウンロードし（ステップ 3）、実行する。

#### 【0187】

次に、図 39 のフローチャートを参照してローダーの処理手順について説明する。

#### 【0188】

ユーザが複合機におけるアプリ切り替えのための所定のボタンを押すことにより、ローダーは J a v a（登録商標）アプリが既にロードされているか否かをチェックし（ステップ 11）、ロードされていれば J a v a（登録商標）アプリを実行する（ステップ 12）。そして、J a v a（登録商標）アプリユーザーインタフェースが表示される。ロードされていなければローダーの画面が表示される（ステップ 13）。なお、J a v a（登録商標）アプリの実行が終了した場合にはローダーの画面が表示される。また、J a v a（登録商標）アプリのユーザーインタフェースが表示されているときに、所定のコードを入力して所定のボタンを押すことにより、ローダーの画面を表示することができる。

#### 【0189】

図 40 に、複合機に表示されるローダーの画面を示す。ローダーには、現在ロードされている J a v a（登録商標）アプリがあればその J a v a（登録商標）アプリと、ローダーが J a v a（登録商標）アプリの更新を W e b サーバに対してチェックする頻度が表示される。

#### 【0190】

ローダーの画面において、アプリのロードのボタンに触れると、図 41 に示すアプリロードウィンドウが表示される（ステップ 14）。ここで J a v a（登録商標）アプリの URL を入力する。URL を入力するには、URL ボタンに触れ、ソフトキーボードを表示させ、ソフトキーボードからタイプすることにより URL を入力する。また、URL ボタンに触れることにより、図 42 に示すような J a v a（登録商標）アプリの URL のリストを表示し、そのリストの中から所

望の J a v a (登録商標) アプリを選択するようにしてもよい。また、必要に応じて、図 4 1 に示すクラス名の欄に、選択した J a v a (登録商標) アプリ (J a r ファイル) の中の m a i n ルーチンを持つクラス名を入力する。

#### 【0191】

その後、図 4 1 に示す画面の O K ボタンに触れることにより、J a v a (登録商標) アプリが W e b サーバからダウンロードされ (ステップ 1 5)、複合機で実行される。

#### 【0192】

ローダーは自動的に、指定された間隔で W e b サーバをチェックし、新たなバージョンの J a v a (登録商標) アプリが W e b サーバに格納されているか否かを確認する。これは、現在動作している J a v a (登録商標) アプリの日付とタイムスタンプを、W e b サーバ上のものと比較することにより行う。新たなバージョンの J a v a (登録商標) アプリが発見された場合には、現在作動している J a v a (登録商標) アプリは削除され、新しいバージョンの J a v a (登録商標) アプリがダウンロードされる。

#### 【0193】

上記の更新間隔を変更するには、図 4 0 に示す画面上の更新間隔ボタンに触れることにより、図 4 3 に示す画面が表示される (ステップ 1 6)。この画面の中で矢印キーを用いて時間か分かを選択し、オペレーションパネルの数字キーを用いて新たな時間間隔を入力する。

#### 【0194】

複合機から J a v a (登録商標) アプリを削除する場合には図 4 0 に示す画面のアンロードのボタンに触れることにより、図 4 4 に示す画面を表示する (ステップ 1 7)。ここで “O K” ボタンに触れることにより J a v a (登録商標) アプリが削除される。

#### 【0195】

図 4 0 の画面における E x i t ボタンに触れることによりローダーが終了する。

#### 【0196】

さて、本実施の形態では、J a v a（登録商標）アプリを連結することにより、カスタマイズプログラムを作成することが可能である。この処理の流れについて図45を参照して説明する。なお、カスタマイズプログラムを作成するプログラム自体もJ a v a（登録商標）アプリであり、以下これをカスタマイズアプリと呼ぶ。なお、カスタマイズアプリは予め画像形成装置内に備えられていてもよい。

#### 【0197】

まず、操作表示部の画面を図40のローダ画面に切り替え、カスタマイズアプリをロードし、実行する。

#### 【0198】

カスタマイズアプリは図45の画面1を表示する。画面1には、紙（paper）のオブジェクトだけが表示される。なお、紙のオブジェクト以外のオブジェクトを入れることもできる。オブジェクトは“入れる”ボタン、“取り出す”ボタンにより出し入れをすることができる。また、画面1において、紙のオブジェクトには読み書き許可属性などの属性を“読書許可”ボタンにより設定することができる。

#### 【0199】

画面1においてpaperに触れると、画面2に示すように、paperオブジェクトに接続されるオブジェクトのリストが表示される。ここでscanを選択すると、画面3に遷移し、さらにscanに触れることにより、scanオブジェクトに接続されるオブジェクトのリストが画面4のように表示される。画面4においてmailを選択すると画面5に遷移する。

#### 【0200】

その後、“設定”ボタンに触れることにより、スキャンして、その画像ファイルをメールにて送信するという処理を行うカスタマイズプログラムが生成される。また、画面6に示すようにキーがアサインされる。複合機でこのキーを操作することにより、上記のカスタマイズプログラムが実行される。このカスタマイズプログラムは、連結されたオブジェクトのクラスを順次実行するように記述されたプログラムである。



**【0201】**

なお、本実施の形態のようにWebサーバにアプリを格納しておき、Webサーバから複合機に所望のアプリをダウンロードすることや、アプリを連結してカスタマイズすることは、他の実施の形態にも適用可能である。例えば、VBインタープリタを用いた前述した構成において、本実施の形態と同様の機能のローダーを備えることにより、本実施の形態と同様にサーバからアプリをダウンロードできる。

**【0202】**

なお、これまでに説明した各サービスプログラム、アプリケーションプログラムは、ICカード等の記録媒体に格納し、そこから画像形成装置に格納することができる。また、ネットワークを介して外部のサーバから画像形成装置に格納することもできる。

**【0203】**

上記のように、本実施の形態によれば、Java（登録商標）環境を用いることにより効率良くアプリを開発することが可能となる。

本発明は、上記の実施の形態に限定されることなく、特許請求の範囲内で種々変更・応用が可能である。

**【発明の効果】**

上記のように本発明によればアプリケーションプログラム開発効率を向上させることができる情報処理装置を提供することができる。

**【図面の簡単な説明】****【図1】**

実施の形態1にかかる複合機の主要構成およびネットワーク構成を示すブロック図である。

**【図2】**

実施の形態1の複合機の機能的構成を示すブロック図である。

**【図3】**

実施の形態1にかかる複合機のハードウェア構成を示すブロック図である。

**【図4】**

リクエストメッセージを受信した複合機における処理の手順を示すフローチャートである。

【図 5】

debug.cgi の処理の手順を示すフローチャートである。

【図 6】

プログラミング画面において、保存ボタン、実行設定ボタン、テスト実行ボタンのいずれかのクリック操作が行われた場合のWEBアプリの処理の手順を示すフローチャートである。

【図 7】

プログラミング画面の内容の一例を示す説明図である。

【図 8】

プログラム入力フィールドに入力されるカスタイズプログラムの一例を示す説明図である。

【図 9】

カスタイズディレクトリに格納されたカスタイズプログラムの名称の一例を示す説明図である。

【図 10】

キー対応付けテーブルの内容の一例を示す説明図である。

【図 11】

複合機の電源投入が行われてから、カスタイズプログラムの実行が可能となるまでの一連の処理の手順を示すフローチャートである。

【図 12】

起動設定ファイルの一例を示す説明図である。

【図 13】

実施の形態 2 にかかる複合機における主要構成を示すブロック図である。

【図 14】

プログラミング画面の一例を示す説明図である。

【図 15】

実施の形態 3 にかかる複合機の機能的構成を示すブロック図である。

**【図 1 6】**

複合機の電源投入が行われてから、カスタマイズプログラムとボタン（キー）との割り付けが完了するまでの一連の処理の手順を示すフローチャートである。

**【図 1 7】**

起動設定ファイルの内容の一例を示す説明図である。

**【図 1 8】**

キー割り付け設定画面の一例を示す説明図である。

**【図 1 9】**

実施の形態 4 にかかる複合機の主要構成およびネットワーク構成を示すブロック図である。

**【図 2 0】**

実施の形態 4 にかかる複合機の機能的構成を示すブロック図である。

**【図 2 1】**

起動設定ファイルの内容の一例を示す説明図である。

**【図 2 2】**

実施の形態 5 にかかる複合機の機能的構成を示すブロック図である。

**【図 2 3】**

PC 2 0 0 上で V i s u a l B a s i c 開発環境を用いて V B アプリを作成している場面を示す図である。

**【図 2 4】**

V B インタプリタにより実行される V B アプリの例を示す図である。

**【図 2 5】**

V B インタプリタにより実行される V B アプリの例を示す図である。

**【図 2 6】**

実施の形態 6 にかかる複合機の機能的構成を示すブロック図である。

**【図 2 7】**

J a v a （登録商標）開発環境と J a v a （登録商標）実行環境を示す図である。

**【図 2 8】**

J a v a（登録商標）実行環境の構成の例を示す図である。

【図 2 9】

Webサーバから J a v a（登録商標）アプリをダウンロードする構成を示す図である。

【図 3 0】

J a v a（登録商標）アプリ作成の手順を示す図である。

【図 3 1】

エミュレータの他の実装例を示す図である。

【図 3 2】

エミュレータにより P C に表示されるオペレーションパネルの画面を示す図である。

【図 3 3】

エミュレータがローダーとして動作する場合に表示される画面を示す図である。

【図 3 4】

操作パネルクラスの階層構成を示す図である。

【図 3 5】

ウィンドウの作成を説明するための図である。

【図 3 6】

サンプルプログラムを示す図である。

【図 3 7】

サンプルプログラムの実行結果を示す図である。

【図 3 8】

図 2 9 に示した構成における J a v a（登録商標）アプリのアップロードからダウンロードまでの概念図である。

【図 3 9】

ローダーの処理手順を示すフローチャートである。

【図 4 0】

複合機に表示されるローダーの画面を示す図である。

**【図 4 1】**

アプリをロードするための画面である。

**【図 4 2】**

J a v a（登録商標）アプリの URL のリストである。

**【図 4 3】**

更新間隔を変更するための画面である。

**【図 4 4】**

アプリをアンロードするための画面である。

**【図 4 5】**

J a v a（登録商標）アプリを連結することにより、カスタマイズプログラムを作成する手順を説明するための図である。

**【符号の説明】**

1 0 0, 1 1 0 0, 1 3 0 0, 1 7 0 0, 1 8 0 0, 1 9 0 0 複合機

1 0 1 白黒ラインプリンタ

1 0 2 カラーラインプリンタ

1 0 3 ハードディスク装置（HDD）

1 0 4 ハードウェアリソース

1 0 5 共有メモリ

1 0 6 h t t p d

1 1 0 ソフトウェア群

1 1 1 プリンタアプリ

1 1 2 コピーアプリ

1 1 3 ファックスアプリ

1 1 4 スキャナアプリ

1 1 5 ネットファイルアプリ

1 1 6 工程検査アプリ

1 1 7 外部アプリ

1 2 0 プラットホーム

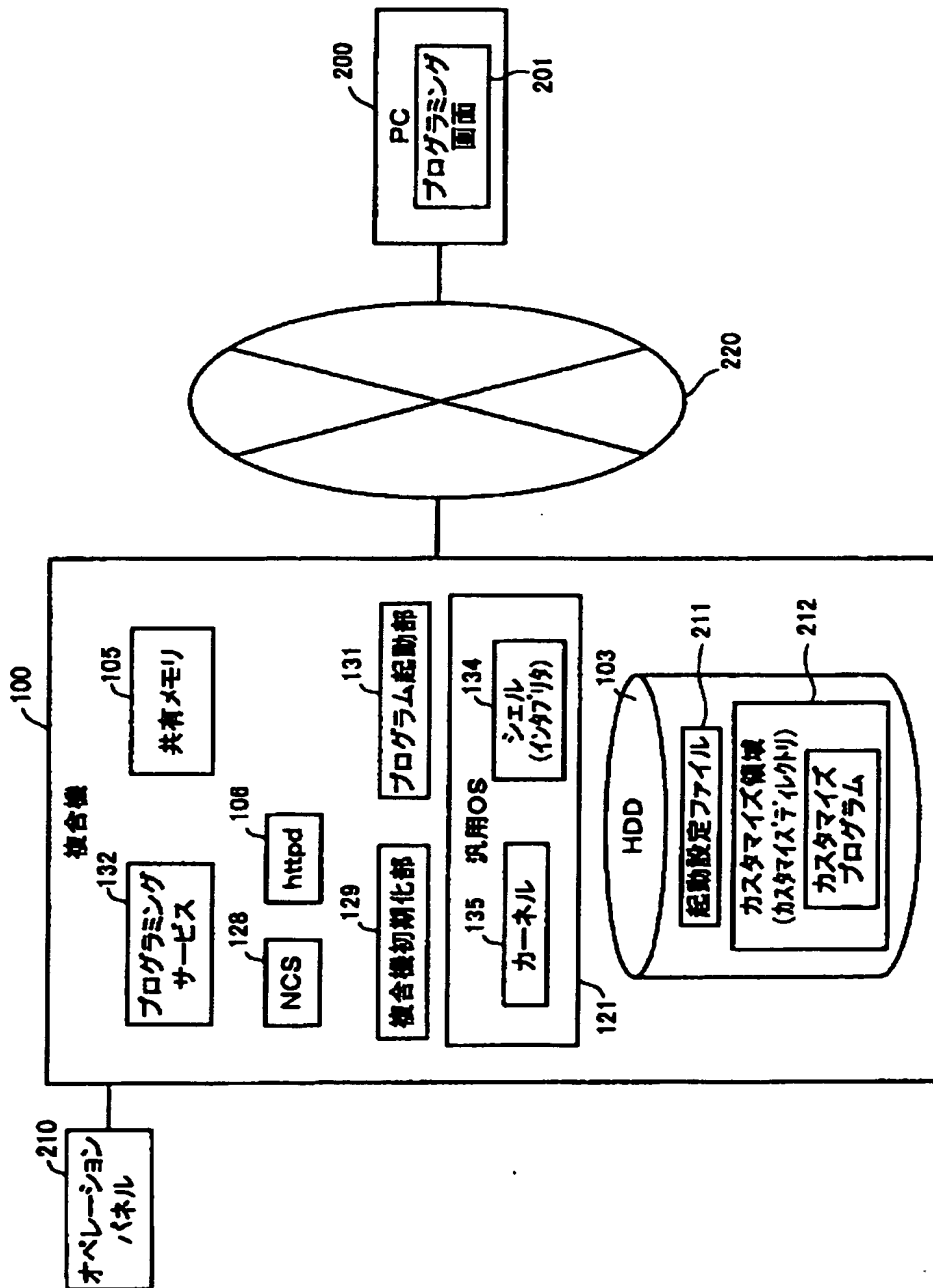
1 2 1 汎用 OS

1 2 2 S C S  
1 2 3 S R M  
1 2 4 E C S  
1 2 5 M C S  
1 2 6 O C S  
1 2 7 F C S  
1 2 8 N C S  
1 2 9 複合機初期化部  
1 3 0 アプリケーション  
1 3 1 プログラム起動部  
1 3 2 , 1 1 3 2 プログラミングサービス  
1 3 3 オペパネ登録部  
1 3 4 インタプリタ (シェル)  
1 3 5 カーネル  
2 0 0 P C 2 0 1 , 1 1 0 1 プログラミング画面  
2 1 1 起動設定ファイル  
2 1 2 カスタマイズディレクトリ  
2 2 0 インターネット  
1 7 3 4 , 1 8 0 1 V B インタプリタ  
1 9 0 1 J a v a (登録商標) 実行環境  
1 9 1 1、1 9 2 1 クラスライブラリ  
1 9 1 2、1 9 2 2 仮想マシン  
1 9 1 4、1 9 2 4 アプリケーション管理部  
1 9 2 3 エミュレータ  
1 9 2 5 J a v a (登録商標) コンパイラ

【書類名】 図面

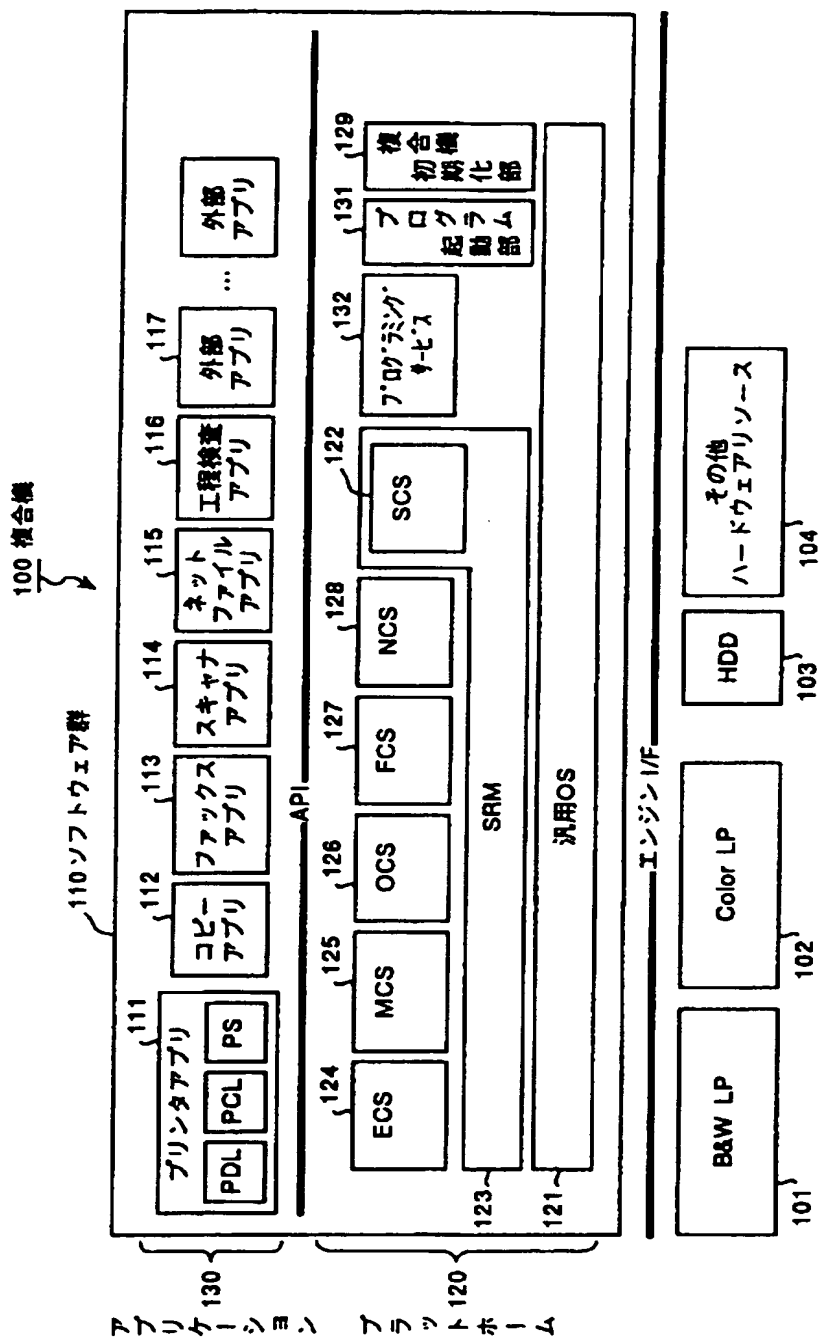
【図 1】

実施の形態1にかかる複合機の主要構成および  
ネットワーク構成を示すブロック図



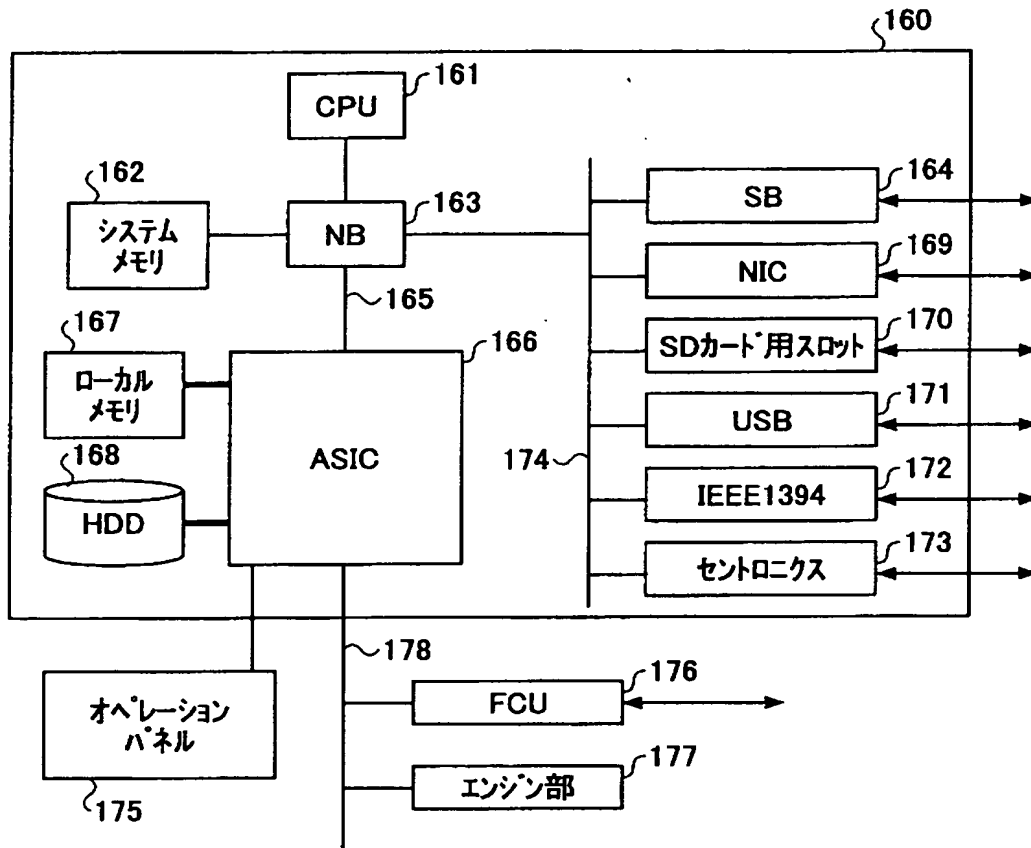
【図 2】

実施の形態1の複合機の機能的構成を示すブロック図



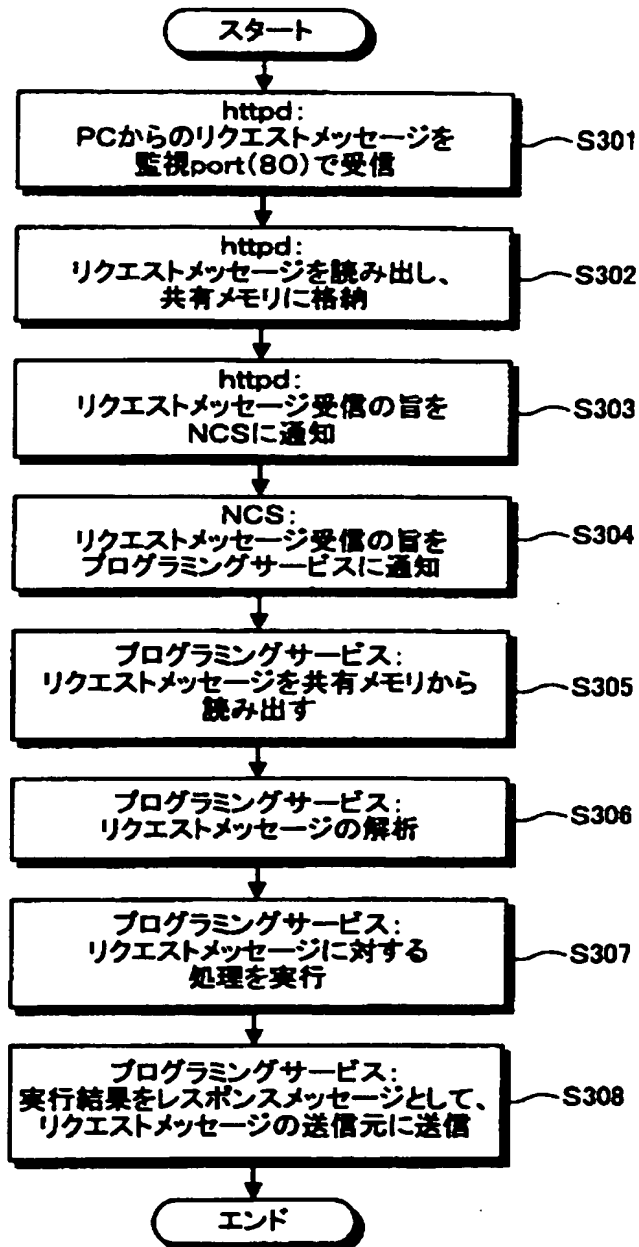


【図 3】

実施の形態1にかかる複合機の  
ハードウェア構成を示すブロック図

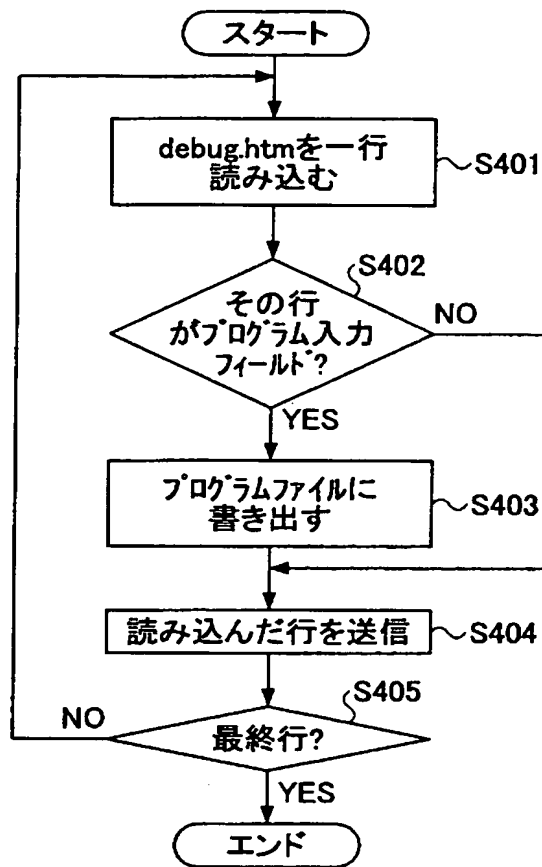
【図 4】

リクエストメッセージを受信した複合機における  
処理の手順を示すフローチャート



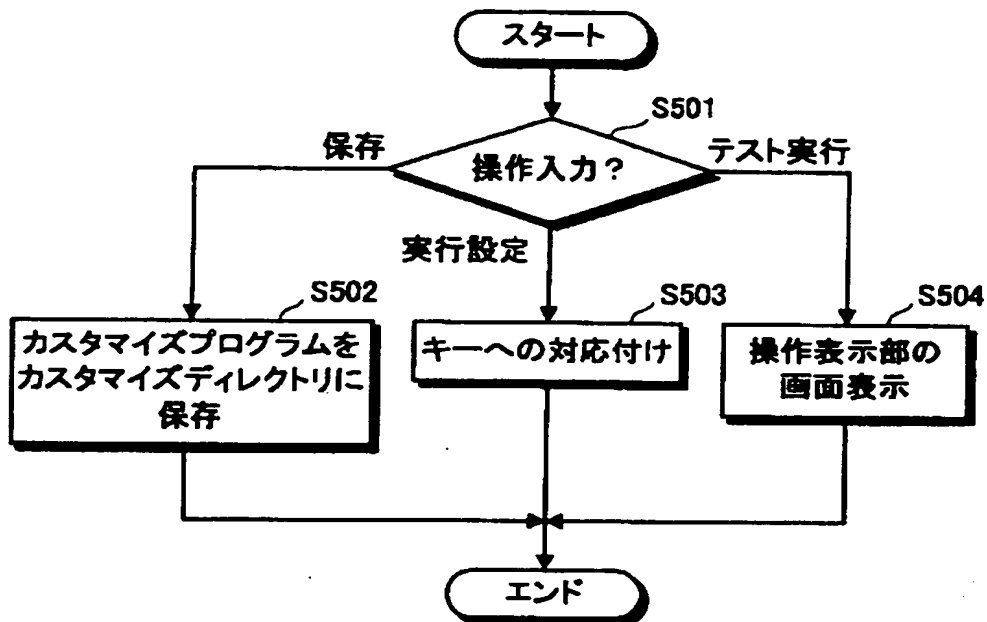
【図 5】

debug.cgiの処理の手順を示すフローチャート



【図 6】

プログラミング画面において、保存ボタン、実行設定ボタン、テスト実行ボタンのいずれかのクリック操作が行われた場合のWEBアプリの処理の手順を示すフローチャート



【図 7】

プロムラミング画面の内容の一例を示す説明図

**プログラミング画面**

参照
アップロード

プログラム入力フィールド

保存
消去
テスト実行
実行設定

【図 8】

プロムラミング入力フィールドに入力される  
カスタマイズプログラムの一例を示す説明図

```
# ocr shell script
#
scanimage -out /hdd/ts/tmp/tmpimg.tif
decomp -in /hdd/ts/tmp/tmpimg.tif -out /hdd/ts/tmp/tmpimg.bmp
ocr -in /hdd/ts/tmp/tmpimg.bmp -out /hdd/ts/tmp/ocrresult.txt
mail xxxxx@yy.zz.co.jp
```

【図 9】

カスタマイズディレクトリに格納された  
カスタマイズプログラムの名称の一例を示す説明図

カスタマイズディレクトリ/ファイル名

hdd/xxx/opepane/shell1  
shell2  
shell3

【図 10】

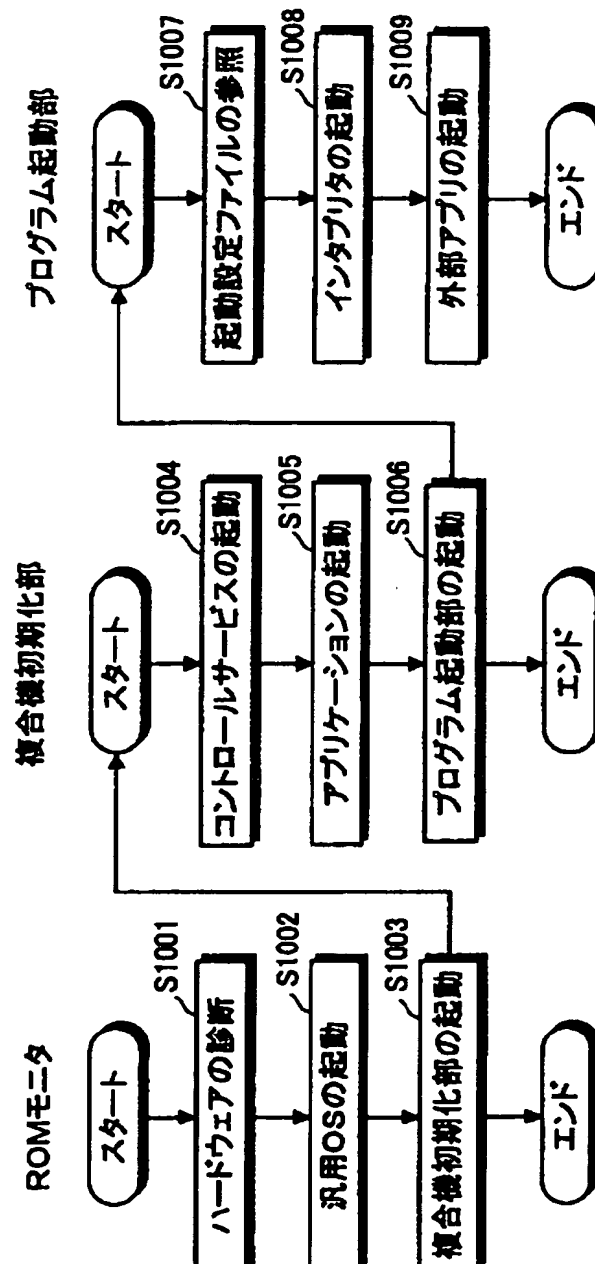
キー対応付けテーブルの内容の一例を示す説明図

キー対応付けテーブル

キーコード:	カスタマイズプログラム
501	: shell1
502	: shell2
503	: shell3

【図 11】

複合機の電源投入が行われてから、カスタマイズプログラムの実行が可能となるまでの一連の処理の手順を示すフローチャート



【図12】

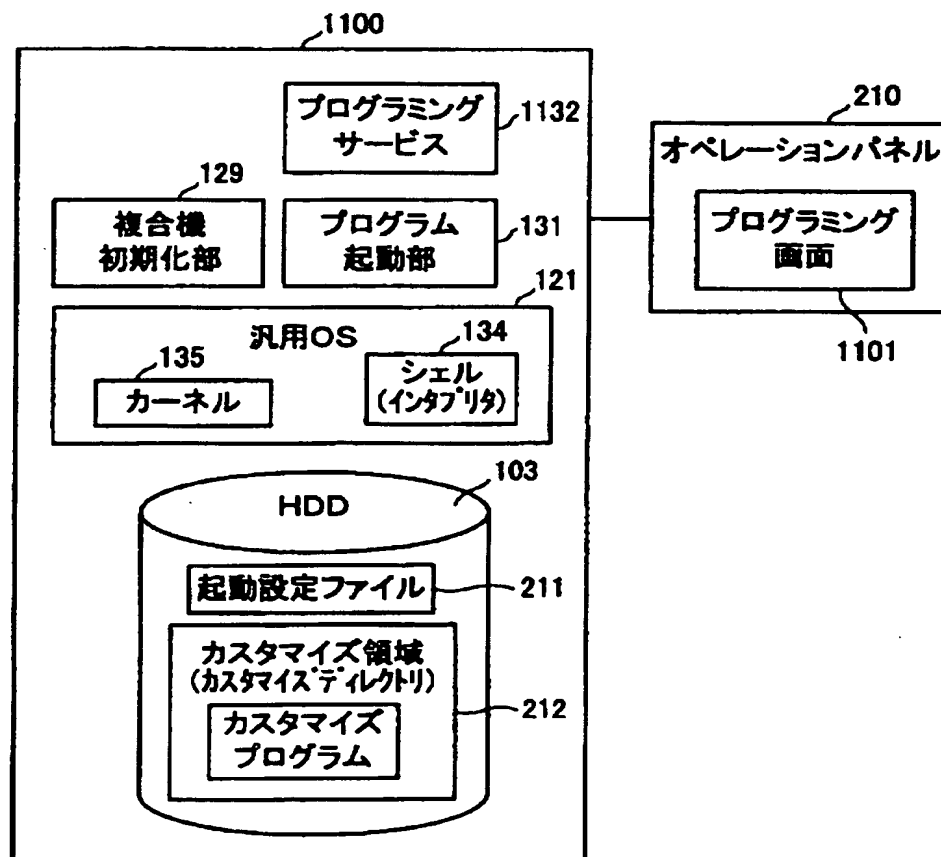
起動設定ファイルの一例を示す説明図

起動設定ファイル

プログラム名称	インタプリタ(シェル)
プログラム名称	XXXアプリ
	⋮
	⋮

【図13】

実施の形態2にかかる複合機における主要構成を示すブロック図





【図 14】

プログラミング画面の一例を示す説明図

プログラミング画面

参照

プログラム入力フィールド

保存

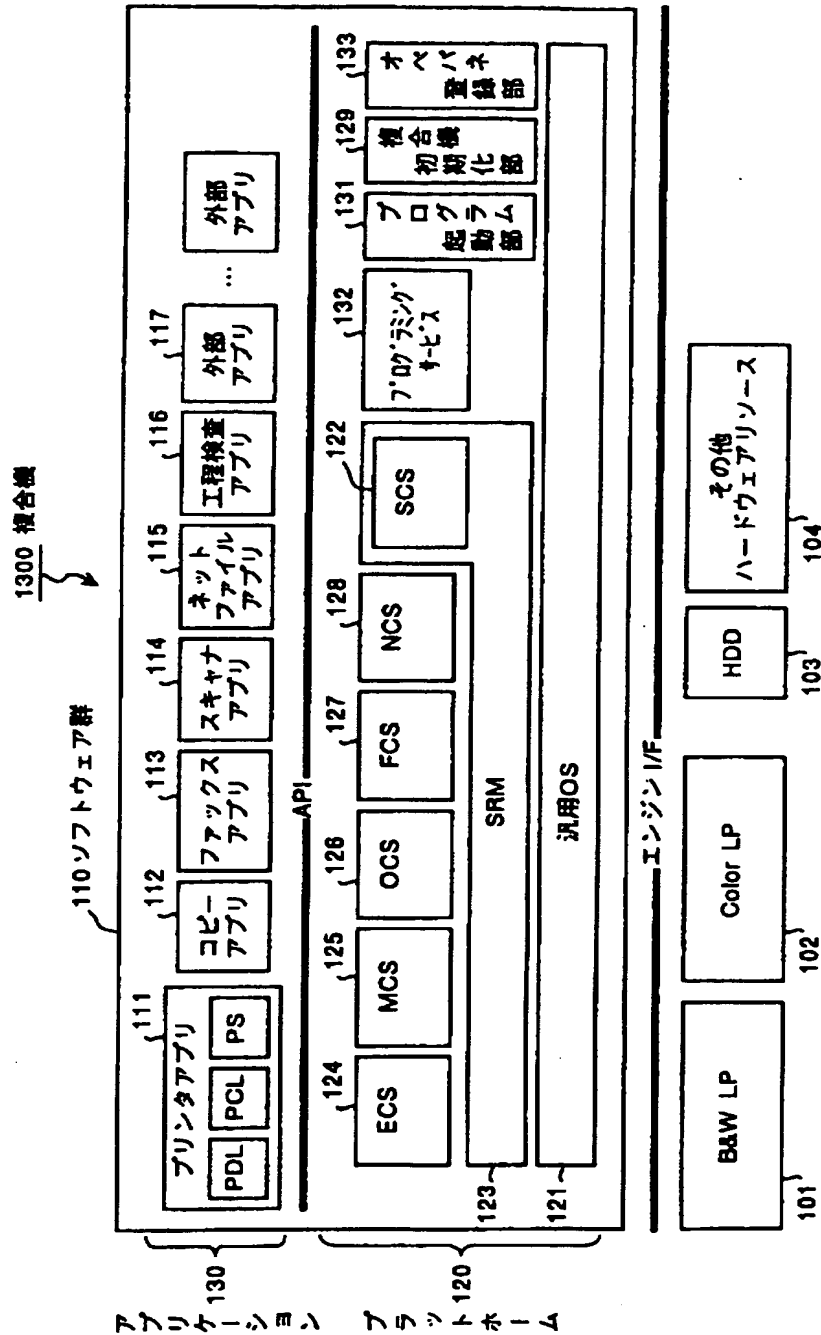
消去

テスト実行

実行設定

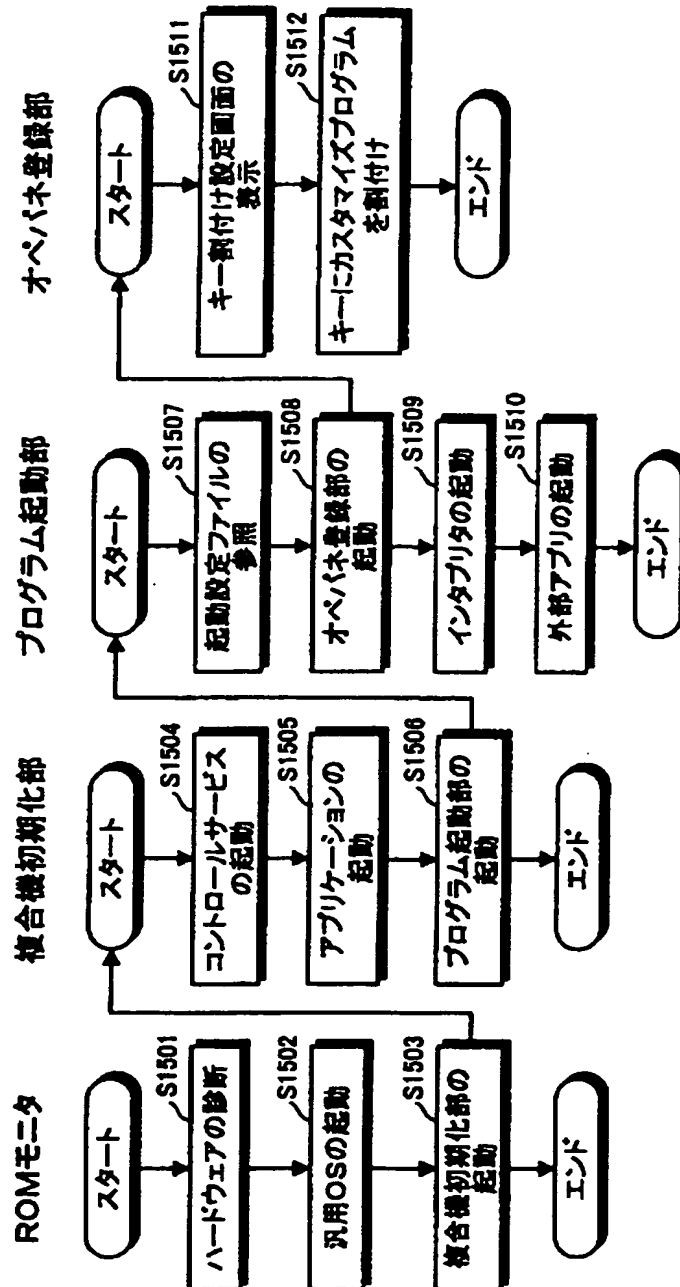
【図 15】

実施の形態3にかかる複合機の機能的構成を示すブロック図



【図 16】

複合機の電源投入が行われてから、カスタマイズプログラムと  
ボタン(キー)との割り付けが完了するまでの一連の  
処理の手順を示すフローチャート



【図 17】

## 起動設定ファイルの内容の一例を示す説明図

## 起動設定ファイル

プログラム名称	オペバネ登録部プログラム
プログラム名称	インタプリタ(シェル)
プログラム名称	XXXアプリ
	⋮

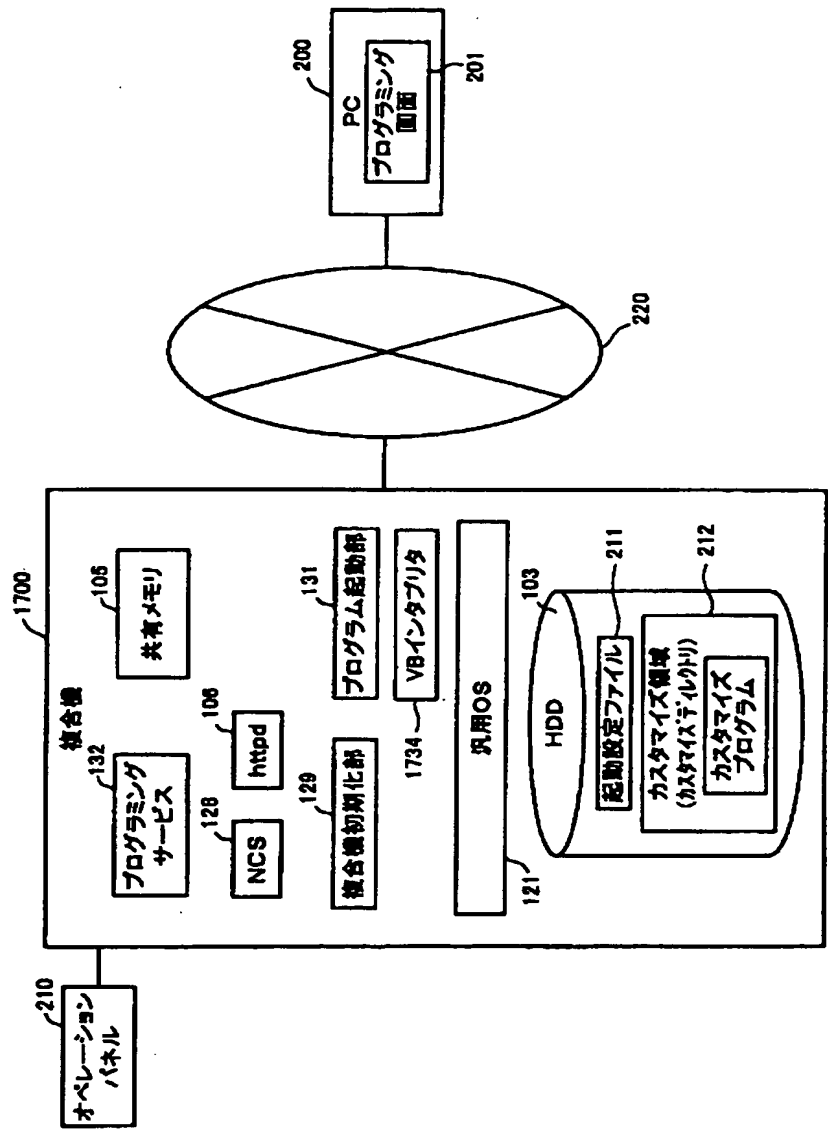
【図 18】

## キー割り付け設定画面の一例を示す説明図

キー割付け設定画面			
カスタマイズプログラム	キー(ボタン)		
Shell1	A	B	C
Shell2	A	B	C
Shell3	A	B	C

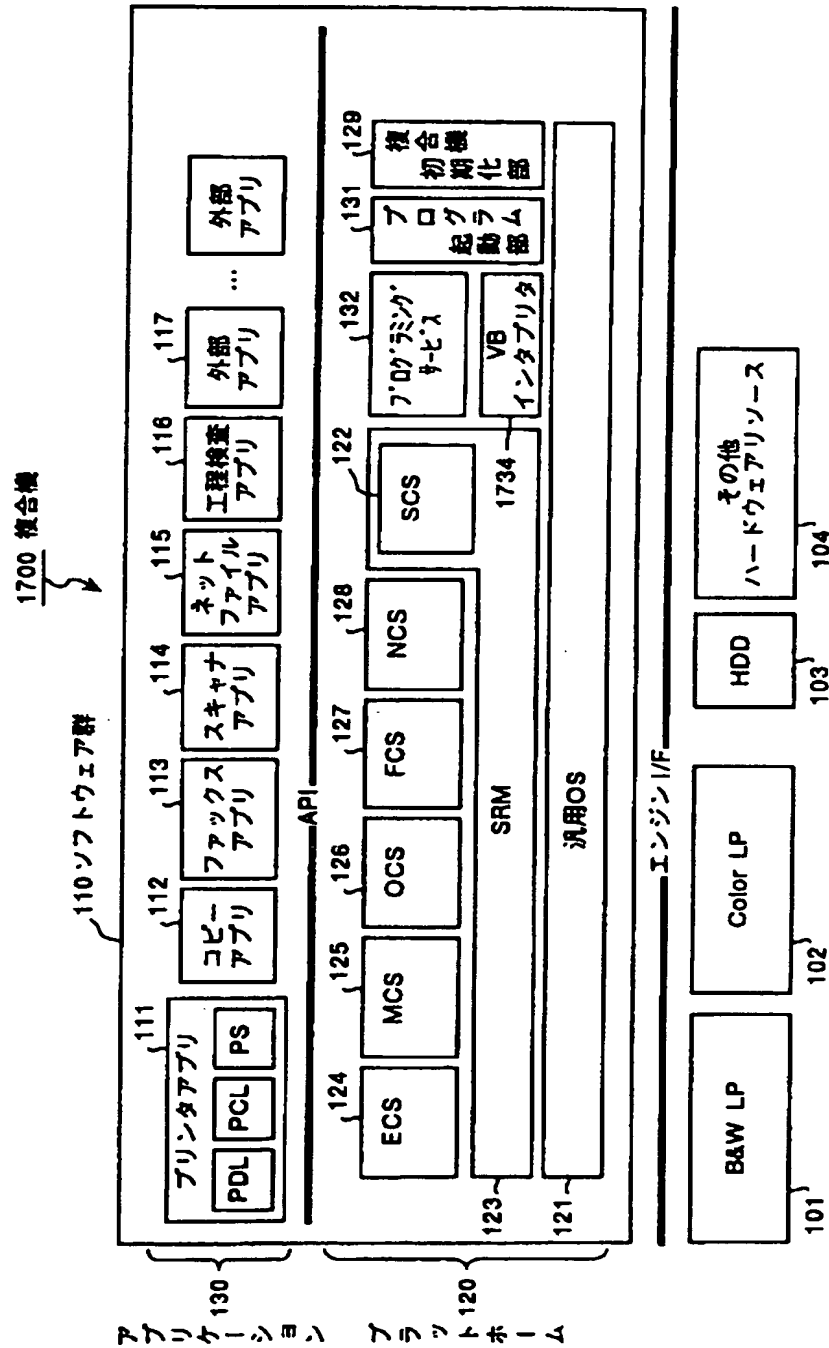
【図 19】

実施の形態4にかかる複合機の主要構成および  
ネットワーク構成を示すブロック図



【図 20】

実施の形態4にかかる複合機の機能的構成を示すブロック図



【図 2 1】

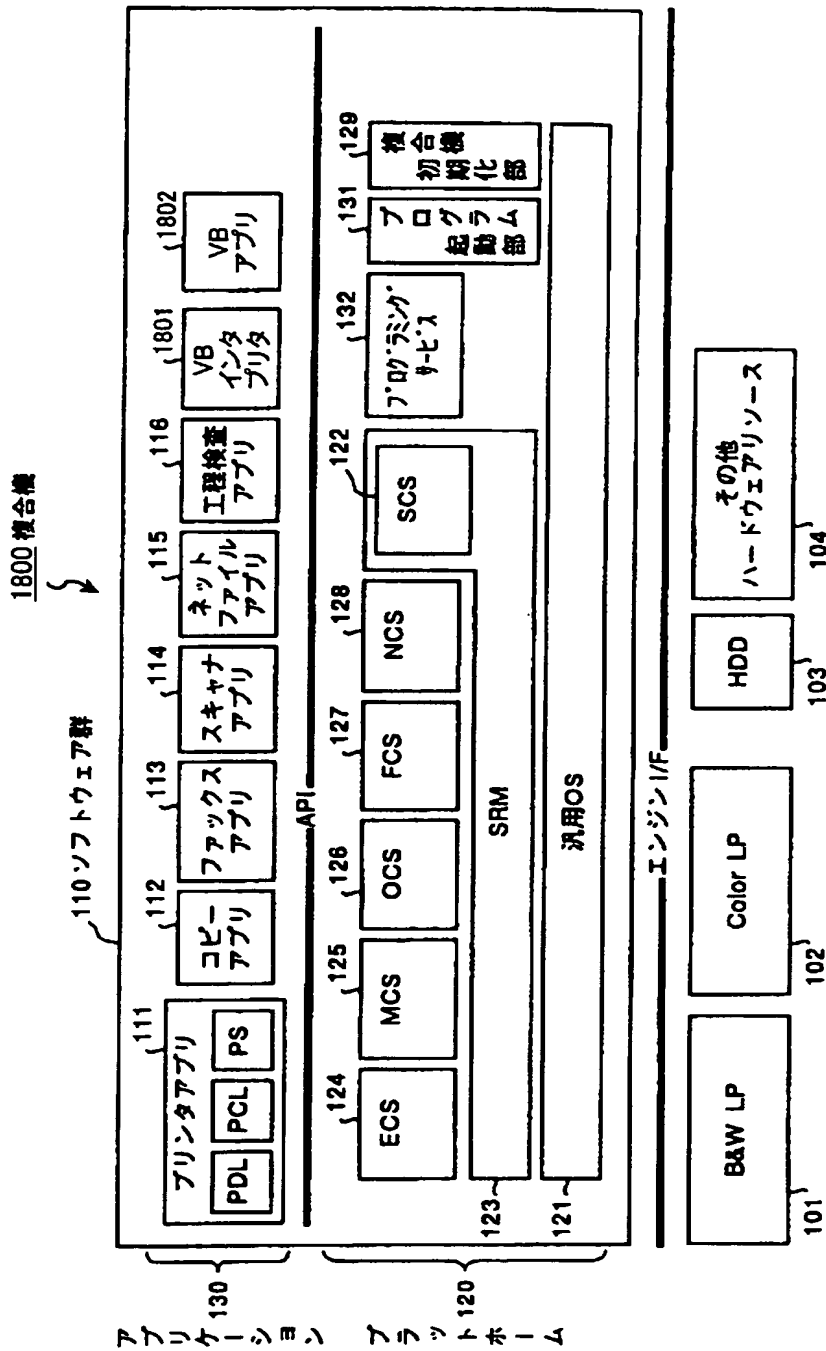
## 起動設定ファイルの内容の一例を示す説明図

## 起動設定ファイル

プログラム名称	VBインタプリタ
プログラム名称	XXXアプリ
	⋮

【図 22】

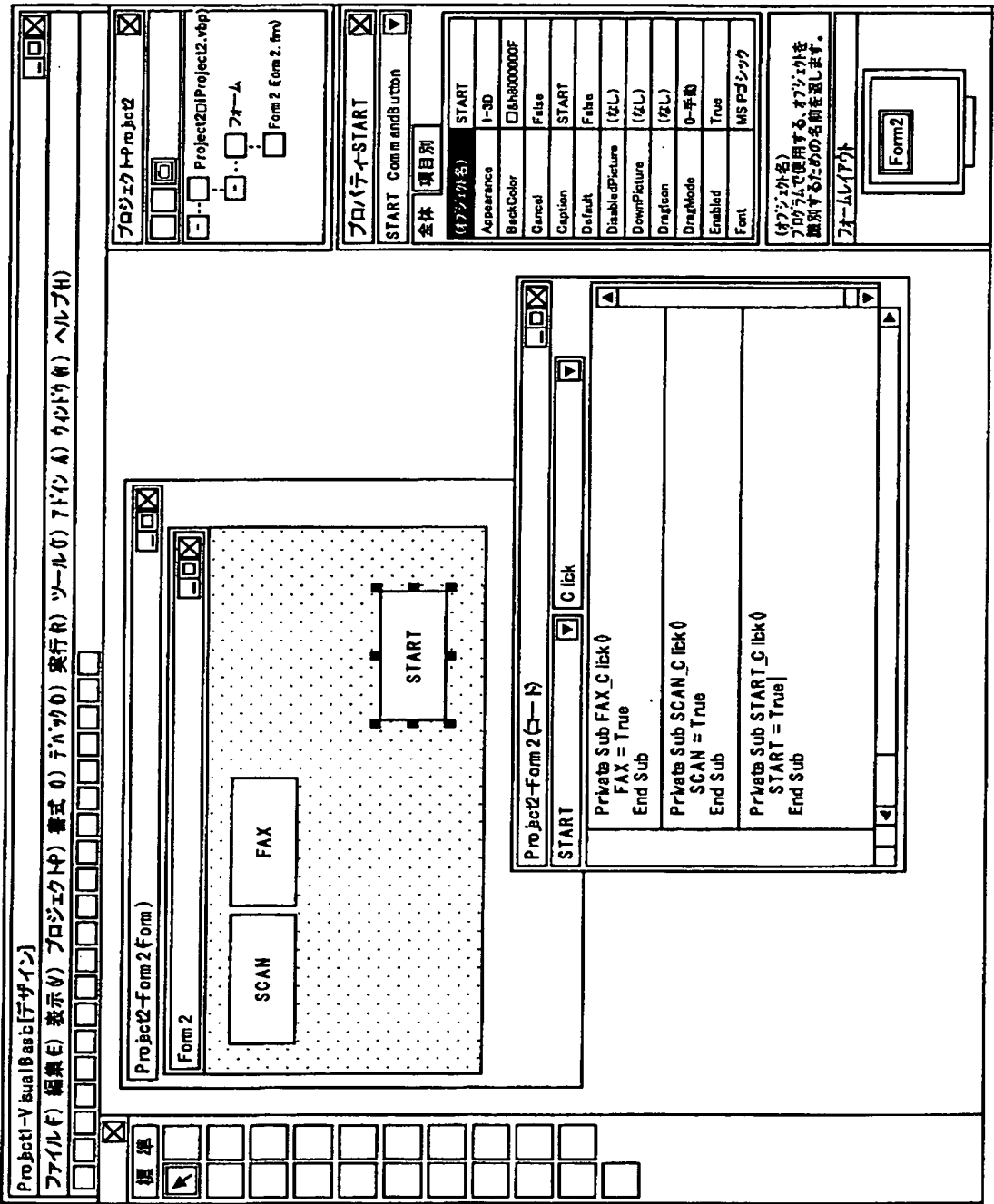
実施の形態5にかかる複合機の機能的構成を示すブロック図





【図 23】

PC200上でVisual Basic開発環境を用いて  
VBアプリを作成している場面を示す図



## 【図 24】

## VBインタプリタにより実行されるVBアプリの例を示す図

Begin VB. Form Form1

```
Caption           = "Form1"
ClientHeight      = 3390
ClientLeft        = 60
ClientTop         = 345
ClientWidth       = 6990
LinkTopic         = "Form1"
ScaleHeight       = 3390
ScaleWight        = 6990
StartPosition     = 3 'Windows の既定値
Begin VB. CommandButton NEXTWIN
```

```
Caption           = "NEXTWIN"
Height            = 735
Left              = 480
TabIndex          = 0
Top               = 480
Width             = 1575
```

End

End

```
Attribute VB_Name = "Form1"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
```

```
Private Sub NEXTWIN_Click()
    /*Form2 ウィンドウへの切り替え */
    call("change_display", "Form2")
End Sub
```

【図 25】

## VBインタプリタにより実行されるVBアプリの例を示す図

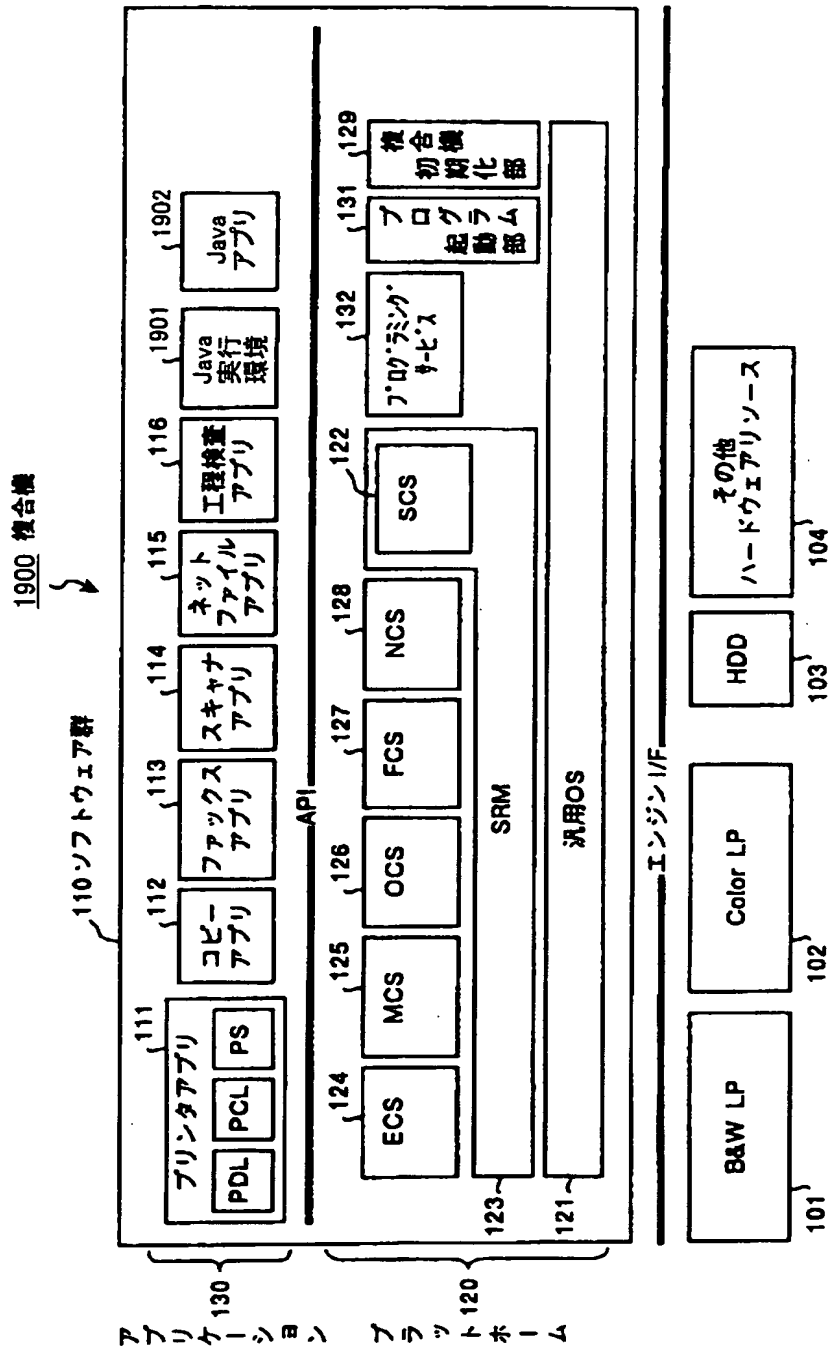
```

Begin VB. Form Form2
    Caption           = "Form2"
    ClientHeight      = 3390
    ClientLeft        = 60
    ClientTop         = 345
    ClientWidth       = 6990
    LinkTopic         = "Form2"
    ScaleHeight       = 3390
    ScaleWight        = 6990
    StartUpPosition   = 3 'Windows の既定値
Begin VB. CommandButton START
    Caption           = "START"
    Height            = 735
    Left              = 4560
    TabIndex          = 2
    Top               = 2040
    Width             = 1815
End
Begin VB. CommandButton FAX
    Caption           = "FAX"
    Height            = 735
    Left              = 2520
    TabIndex          = 1
    Top               = 480
    Width             = 1455
End
Begin VB. CommandButton SCAN
    Caption           = "SCAN"
    Height            = 735
    Left              = 480
    TabIndex          = 0
    Top               = 480
    Width             = 1575
End
End
Attribute VB_Name = "Form1"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Private Sub FAX_Click()
    FAX = True
End Sub
Private Sub SCAN_Click()
    call("ScanStart", A4, ADF, TIFF, BINARY, "/work/tmpfile.tif")
    gwOpItemCreate(win, MESSAGE_ITEM,
        ITEM_MESSAGE_X, 0,
        ITEM_MESSAGE_Y, 12,
        ITEM_MESSAGE_WIDTH, 100,
        ITEM_MESSAGE_HEIGHT, 12,
        ITEM_MESSAGE_CENTER_X, 0,
        ITEM_MESSAGE_BLINK, 0,
        ITEM_MESSAGE, "文字列", 0,
        ITEM_MESSAGE_FRONT, FONT_12, 0,
        0);
End Sub
Private Sub START_Click()
    START = True
End Sub

```

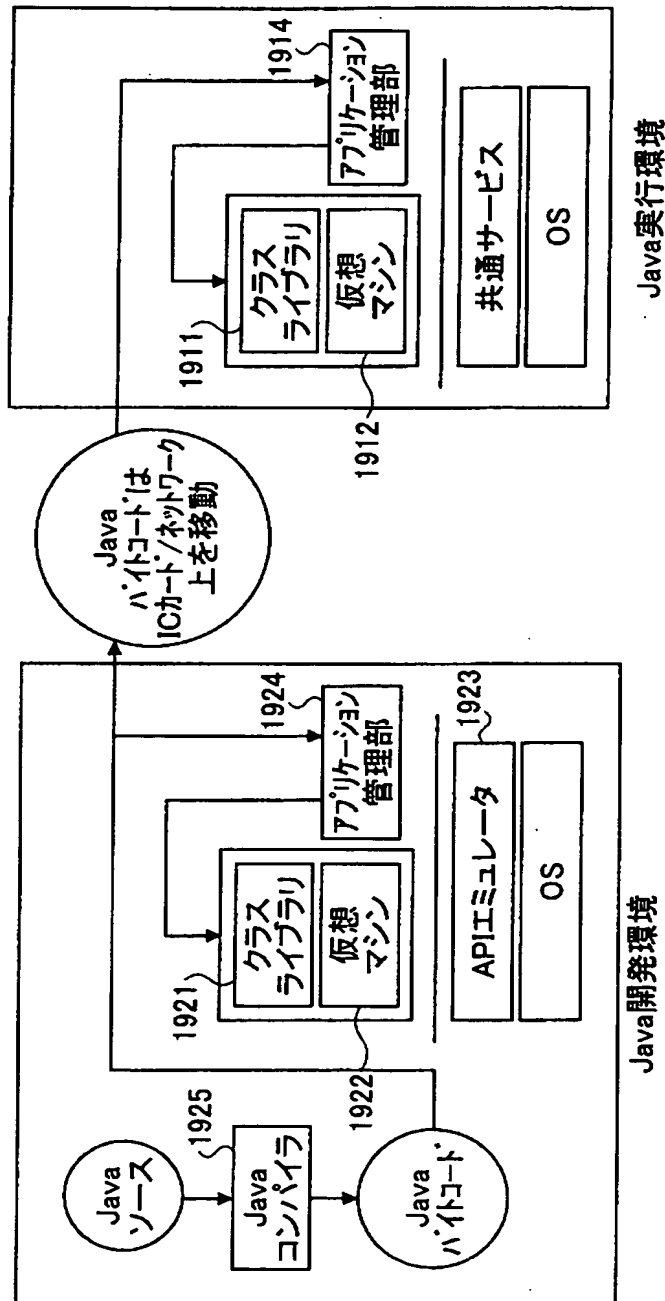
【図 26】

実施の形態6にかかる複合機の機能的構成を示すブロック図



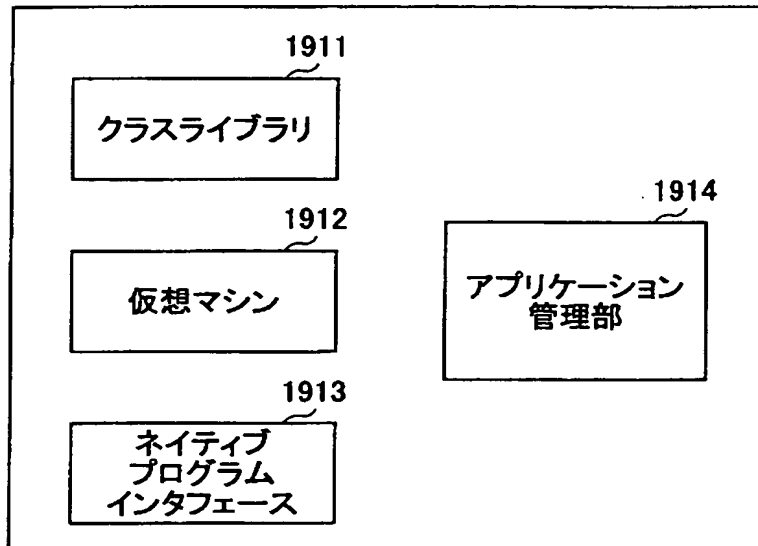
【図 27】

Java(登録商標)開発環境とJava実行環境を示す図



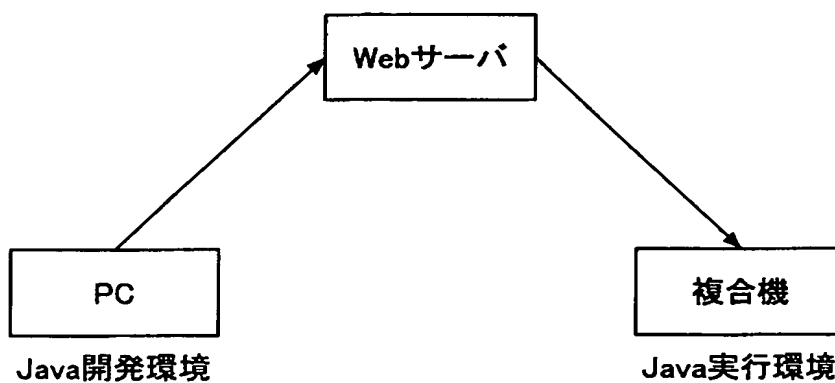
【図 28】

Java(登録商標)実行環境の構成の例を示す図



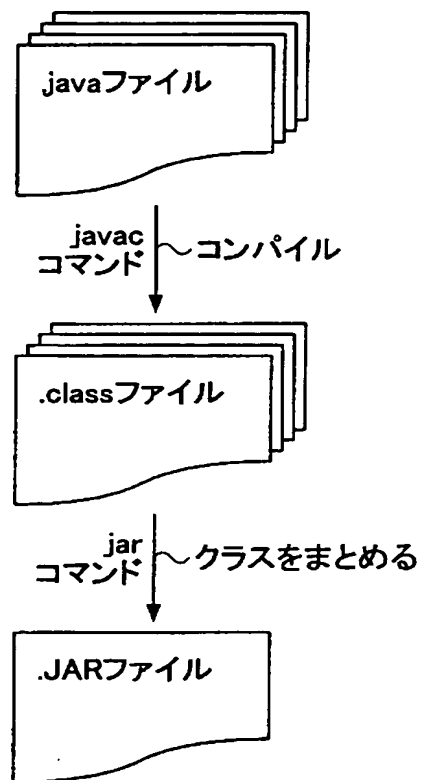
【図 29】

WebサーバからJavaアプリをダウンロードする  
構成を示す図



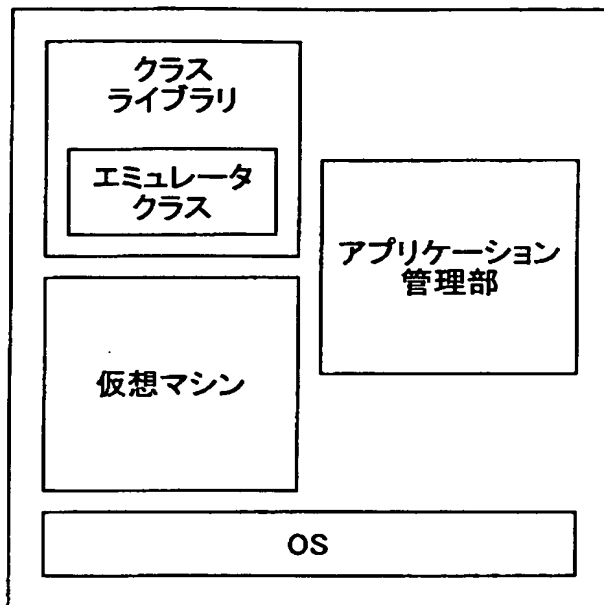
【図 30】

## Javaアプリ作成の手順を示す図



【図 31】

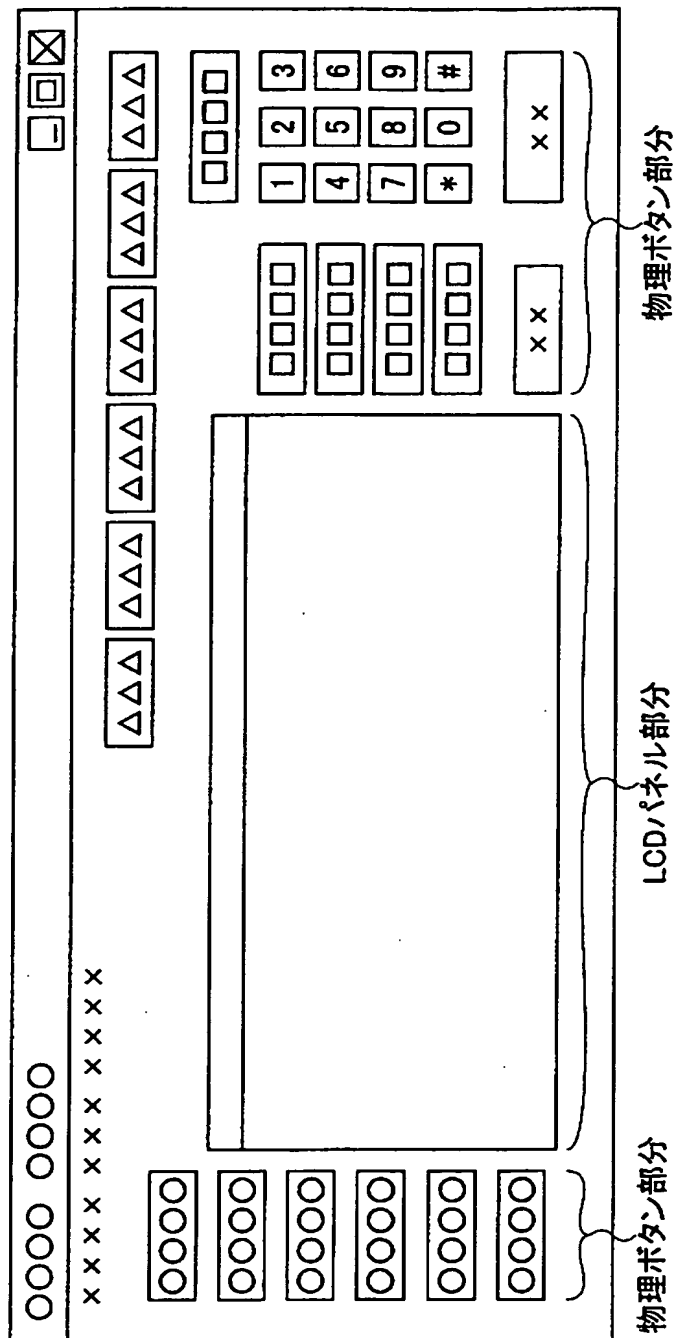
## エミュレータの他の実装例を示す図





【図 32】

エミュレータによりPCに表示される  
オペレーションパネルの画面を示す図



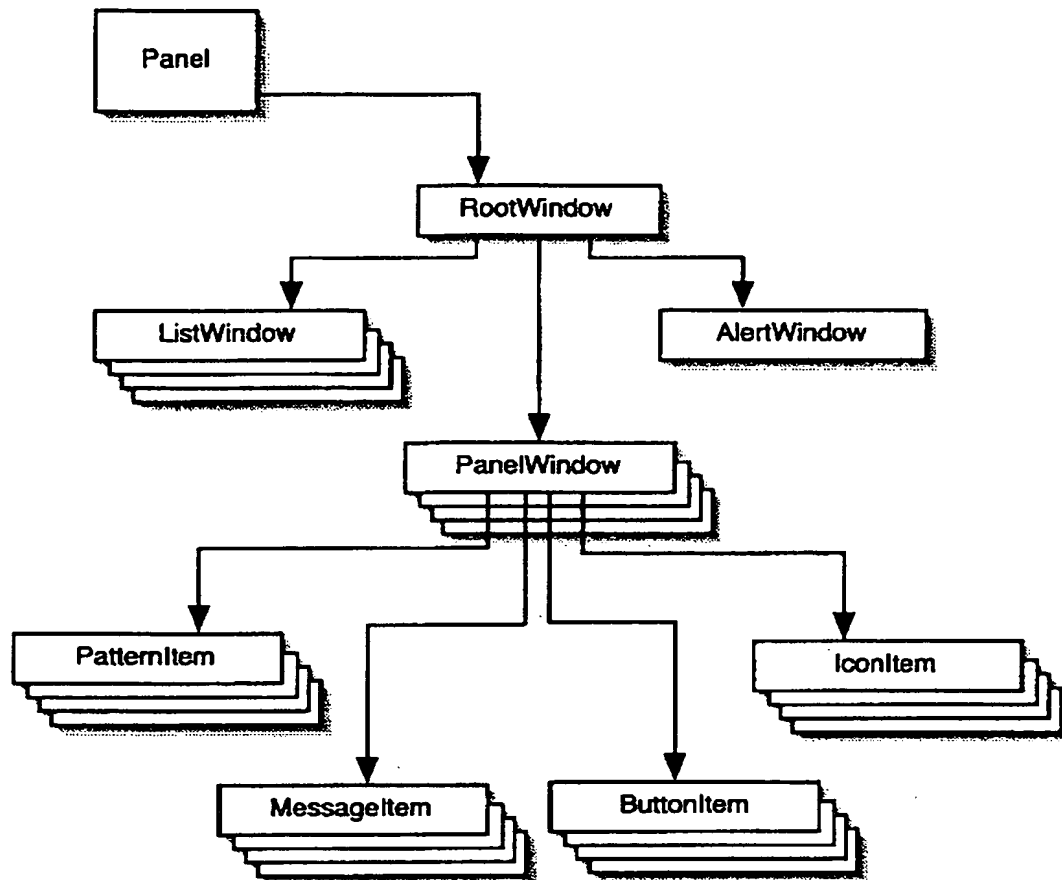
【図 33】

エミュレータがローダーとして動作する場合に表示される画面を示す図

The image shows a graphical user interface for an emulator loader. It consists of a main rectangular frame. Inside this frame, there is a smaller rectangular box in the upper left corner containing three lines of text, each preceded by an unchecked checkbox:   
□ABCアプリ  
□XXXアプリ  
□CCCアプリ  
Below this box, centered horizontally, is another rectangular box containing the text "ABCアプリ". At the bottom right of the main frame, there are two separate rectangular buttons. The left button is labeled "実行" (Execute) and the right button is labeled "キャンセル" (Cancel).

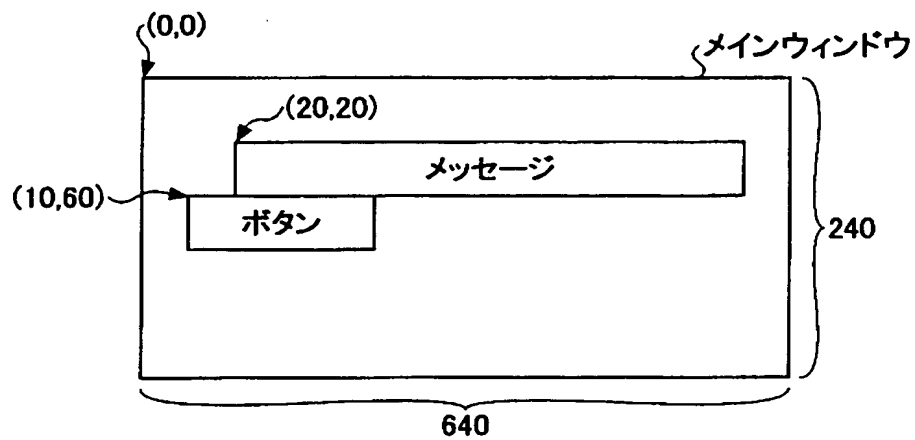
【図 34】

操作パネルクラスの階層構成を示す図



【図 35】

ウィンドウの作成を説明するための図



【図 36】

## サンプルプログラムを示す図

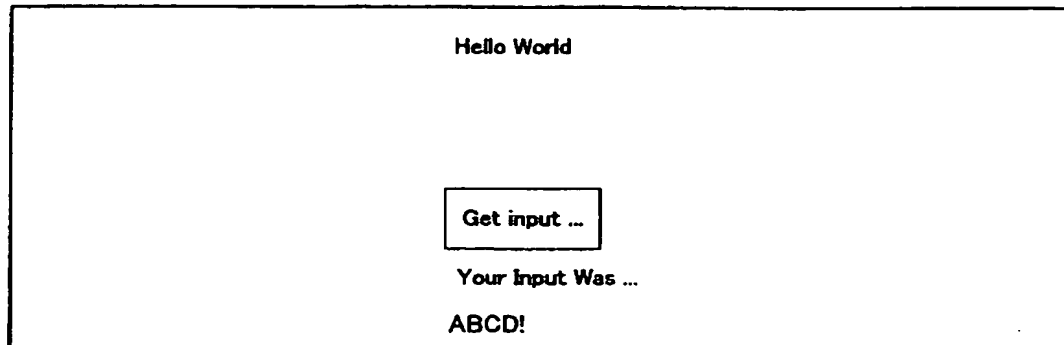
```

import java.lang.*;
import com.hijke.gw.panel.*;           //←①
public class UserApp extends GApp      //←②
{
    public void main()
    {
        Panel panel = new Panel();
        PanelWindow mainwindow =
            new PanelWindow(panel.root());
        ItemString hello = new ItemString("Hello World"); //←③
        MessageItem salutation = new MessageItem(hello);
        salutation.setRect(100, 10, 100, 10);
        mainwindow.addItem(salutation);
        ItemString inputstr =
            new ItemString("Get input...");
        ButtonItem inputButton = new ButtonItem(); //←④
        inputButton.setButtonShape(ButtonItem.BTN_ZAB);
        inputButton.setRect(100, 100, 100, 100);
        inputButton.setWink(true);
        inputButton.addChangeListener(
            new GChangeListener()
            {
                public void stateChanged(GEvent e)
                {
                    String answer =
                        SoftKeyboardWindow.request(
                            "Add your Input",
                            "", 256,
                            ItemString.LANG_ENGLISH_US
                        );
                    inputdisplay.setMessage(
                        new ItemString(answer));
                    inputdisplay.paint();
                }
            }
        );
        mainwindow.addItem(inputButton);
        ItemString label1 =
            new ItemString("Your Input Was..."); //←⑤
        MessageItem label = new MessageItem(label1);
        label.setRect(100, 110, 200, 10);
        mainwindow.addItem(label);
        ItemString outputstr = new ItemString("");
        MessageItem
            inputdisplay = new MessageItem(outputstr); //←⑥
        inputdisplay.setRect(100, 125, 200, 10);
        mainwindow.addItem(inputdisplay);
    }
}

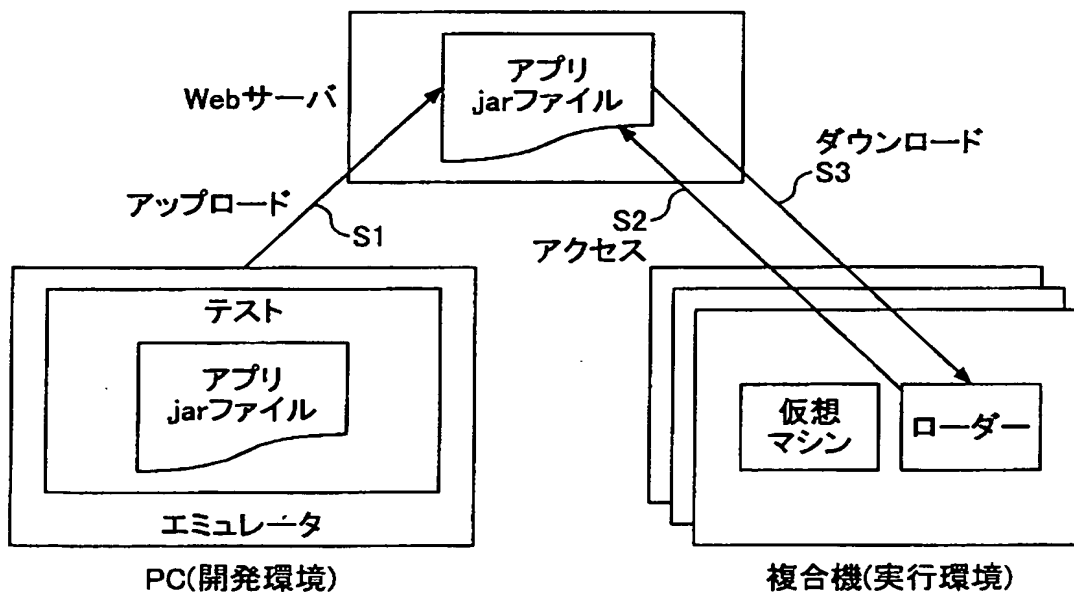
```

【図 37】

サンプルプログラムの実行結果を示す図

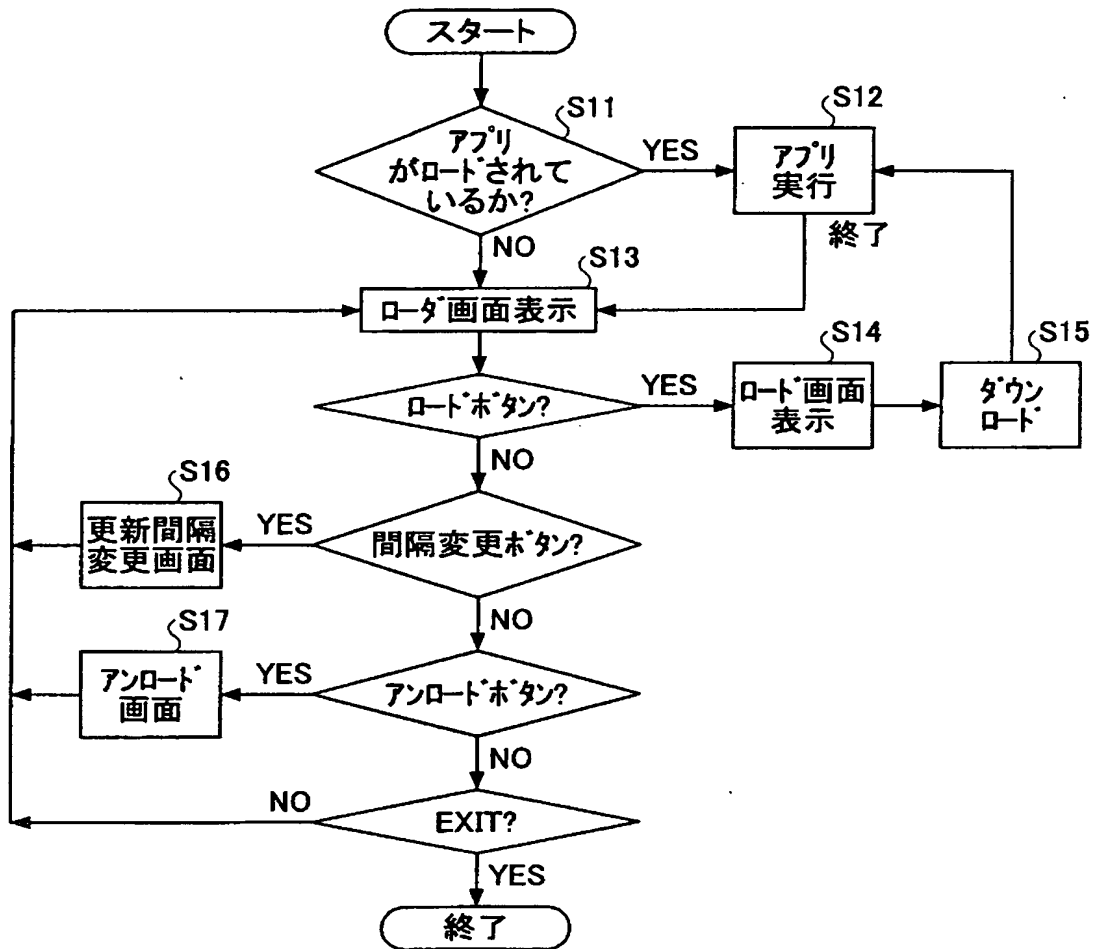


【図 38】

図 29 に示した構成における J a v a アプリの  
アップロードからダウンロードまでの概念図

【図 39】

ローダーの処理手順を示すフローチャート



【図 40】

複合機に表示されるローダーの画面を示す図

アプリケーションローダー		Exit
ロード済アプリ	<input type="text" value="http://xxx.xxx/xxx/abc.jar"/>	
更新間隔	<input type="text" value="002:00"/>	
アプリケーションロード		更新間隔
		アプリケーションアンロード

【図 41】

アプリをロードするための画面

ロードアプリケーション	
URL	<input type="text" value="http://xxx.xxx/xxx/abc.jar"/>
Class Name	<input type="text" value="aaa"/>
OK	
キャンセル	



【図 4 2】

# JavaアプリのURLのリスト

http://xxx.xxx/xx/aaa.jar  
 http://xxx.xxx/xx/bbb.jar  
 http://xxx.xxx/xx/ccc.jar  
 ⋮

【図 4 3】

## 更新間隔を変更するための画面

更新間隔

時間入力して下さい

002:00

<000:10-999:59>

←

→

クリア

OK

キャンセル

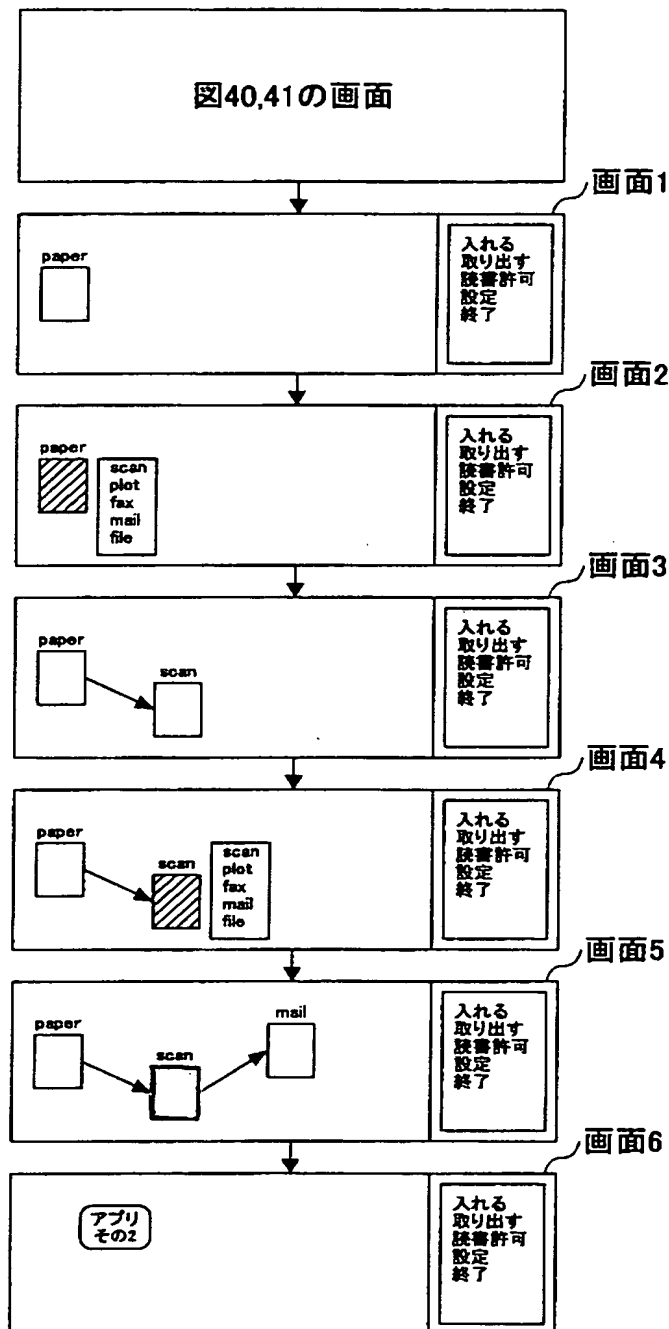
【図 44】

アプリをアンロードするための画面

アプリのアンロード	
現アプリをアンロードしますか?	
OK	キャンセル

【図 45】

Javaアプリを連結することにより、  
カスタマイズプログラムを作成する手順を説明するための図



【書類名】 要約書

【要約】

【課題】 プログラム開発効率を向上させることを目的とする。

【解決手段】 画像形成装置で実行されるプログラムを開発するために使用される情報処理装置に、プログラムを解釈し、実行する実行手段と、当該プログラムの命令に対する画像形成装置の動作に相当する動作を前記情報処理装置に実行させるエミュレータとを備える。

【選択図】 図 27

特願 2 0 0 3 - 1 9 9 9 4 9

出 願 人 履 歴 情 報

識別番号

[ 0 0 0 0 0 6 7 4 7 ]

1. 変更年月日            1 9 9 0 年    8 月 2 4 日  
   [変更理由]            新規登録  
     住 所                東京都大田区中馬込 1 丁目 3 番 6 号  
     氏 名                株式会社リコー
  
2. 変更年月日            2 0 0 2 年    5 月 1 7 日  
   [変更理由]            住所変更  
     住 所                東京都大田区中馬込 1 丁目 3 番 6 号  
     氏 名                株式会社リコー